DETERMINING THE INERTIAL STATES OF LOW PRANDTL NUMBER FLUIDS USING ELECTROCHEMICAL CELLS

By

DANIEL W. CRUNKLETON

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA AS PARTIAL FULFULLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2002

I wish to dedicate this work to the memory of my grandfather, Mr. Brents L. Carter; and my grandmother, Mrs. Bernice Crunkleton.

ACKNOWLEDGMENTS

I wish to extend my deepest appreciation to the members of my supervisory committee for their help and guidance. Special thanks are extended to the chairman, Tim Anderson, and the co-chairman, Ranga Narayanan, for their encouragement. Additionally, the technical expertise (and, most importantly, friendship) of Dr. Gérard Labrosse of the Physics Department at the Université Paris – Sud was irreplaceable.

The many discussions (not all of which were research related!) of several people were greatly appreciated. Several Okies deserve thanks, including John Dunn and Beau Jestice who have always been there for me. The new friends I have made in Florida also deserve specific thanks, including Brian Burgess, Omar Bchir, Steve Johnston, Mushin Ider, and Jianyun Shen. Their different approaches to problems made them invaluable and irreplaceable. I couldn't have asked for better colleagues, confidants, and friends.

Finally, thanks are due to my family for their patience and confidence while I was finishing my studies. Specifically, I wish to thank my parents, Larry and Dianne, for their steadfast support and my uncle, Dr. Richard Carter, who demonstrated that this entire trip through grad school was, indeed, possible.

TABLE OF CONTENTS

LIST OF SYMBOLS

Latin

a	Activity
c	Molar concentration
D	Diffusivity (also known as "Diffusion Coefficient")
E	Electromotive force
f	Arbitrary function
F	Faraday constant (96,487 C/eq)
H	Height
i	$\sqrt{-1}$
I	Current
J	Molar flux
L	Length
n	Valency of an electrochemical reaction
Nu_s	Surface-averaged Nusselt number
Pr	Prandtl number, $Pr = \frac{\nu}{\kappa}$
R	Gas constant (8.314 J/(mol-K))
Ra	Rayleigh number, $Ra = \frac{g \beta_T \Delta T H^3}{\nu \kappa}$
Ra_{c1}	First critical Rayleigh number
Ra_{c2}	Second critical Rayleigh number

Ra_{c3}	Third critical Rayleigh number
Sc	Schmidt number, $Sc = \frac{\nu}{D}$
t	Time
t^*	Dimensionless time, $t^* = \frac{t}{\left(H^2/\nu\right)}$
T	Absolute temperature
T_c	Cold Temperature
T_h	Hot Temperature
T^*	Dimensionless temperature, $T^* = \frac{T-T_c}{T_h-T_c}$
v^*	Dimensionless velocity, $v^* = \frac{v}{\left(\nu/H\right)}$
W	Width
x,y,z	Cartesian directions

<u>Greek</u>

β_T	Thermal expansion coefficient, $\beta_T = -\frac{1}{\rho_o}\frac{\partial\rho}{\partial T}\Big\|_{T_o}$
β_c	Concentration expansion coefficient, $\beta_C = -\frac{1}{\rho_o}\frac{\partial\rho}{\partial c}\Big\|_{c_o}$
γ	Activity coefficient
γ	Aspect ratio, $\gamma = \frac{L}{H}$

κ	Thermal diffusivity
ν	Kinematic viscosity
ρ	Density
τ	Time constant

Subscript/Superscript

$_o$	Condition at which the Boussinesq approximation is taken
*	Dimensionless quantity

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements of the Degree of Doctor of Philosophy

DETERMINING THE INERTIAL STATES OF LOW PRANDTL NUMBER FLUIDS USING ELECTROCHEMICAL CELLS

By

Daniel W. Crunkleton

May 2002

Chairman: Dr. Timothy J. Anderson
Co-Chairman: Dr. Ranganathan Narayanan
Major Department: Chemical Engineering

The quality of crystals grown from the melt is often deteriorated by the presence of buoyancy-induced convection, produced by temperature or concentration inhomogenities. It is, therefore, important to develop techniques to visualize such flows. In this study, a novel technique is developed that uses solid-state electrochemical cells to establish and measure dissolved oxygen boundary conditions. To visualize convection, a packet of oxygen is electrochemically introduced at a specific location in the melt. As the fluid convects, this oxygen packet follows the flow, acting as a tracer. Electrochemical sensors located along the enclosure then detect the oxygen as it passes.

Over sufficiently long times, oxygen diffusion is important; consequently, the oxygen diffusivity in the fluid is measured. This diffusivity is determined using both transient and steady state experiments with tin and tin-lead alloys as model fluids. It is concluded that the presence of convection due to solutal gradients and/or tilt increases the measured diffusivity by one-half to one order of magnitude. The oxygen diffusivity in tin-lead alloys is measured and the results show that the alloy diffusivities are lower than those of the corresponding pure metals. This concentration functionality is explained with a multicomponent diffusion model and shows good agreement.

Rayleigh-Bénard convection was used to validate the electrochemical approach to flow visualization. Thus, a numerical characterization of the second critical Rayleigh numbers in liquid tin was conducted for a variety of Cartesian aspect ratios. The extremely low Prandtl number of tin represents the lowest value studied numerically. Additionally, flow field oscillations are visualized and the effect of tilt on convecting systems is quantified.

Finally, experimental studies of the effect of convection in liquid tin are presented. Three geometries are studied: (1) double cell with vertical concentration gradients; (2) double cell with horizontal concentration gradients; and (3) multiple cell with vertical temperature gradients. The first critical Rayleigh number transition is detected with geometry (1) and it is concluded that current measurements are not as affected by convection as EMF measurements. The system is compared with numerical simulations in geometry (2), and oscillating convection is detected with geometry (3).

CHAPTER 1
INTRODUCTION

Low Prandtl-number fluids, such as liquid metals and semiconductor melts, are precursors in several crystal growth processes, including the Bridgman and Czochralski techniques. In these methods, the melt solidifies to form a single crystal material. Fluid motion in the melt, however, can adversely affect the quality of the material grown by these techniques. This arises from buoyancy introduced into the fluid through temperature or concentration gradients. If the fluid is heated from below, the density of the cooler fluid on top is greater than that on bottom. Then, the cooler, heavier fluid is pulled downward under the force of gravity and the warmer fluid flows upward. The same process occurs when a top-heavy solutal gradient is present. The resulting motion is called buoyancy-induced convection or gravity-induced convection and is shown in Figure 1-1 for a right cylindrical geometry.

If this convecting fluid is to be used to grow a crystal, the presence of convection significantly diminishes the quality of the crystal that is produced [Mul88]. Convection can localize the deposition of impurities; therefore, instead of a uniform dopant distribution, the crystal will have localized inhomogenities, diminishing the quality of the crystal. Moreover, if the melt is convecting as a solid is crystallized, the morphology of the crystal is damaged which can produce dislocations in the lattice, which also affects the crystal usefulness. Therefore, a detailed characterization of convection in melts deserves further study.

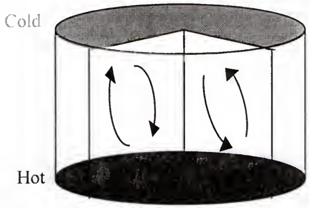


Figure 1- 1. Fluid motion induced by heating from below or by top-heavy solutal gradient (shown for a cylinder, but applicable to any geometry).

The topics examined in this work all relate somehow to this convection which occurs when buoyancy is present in low Prandtl-number fluids either by thermal or solutal gradients (but not both). At this point, however, it is useful to examine the physics of this convection in more detail.

1.1 Critical Rayleigh Numbers

Natural convection that results when fluids are heated from below is a problem that has been studied extensively over the last century. The problem is often referred to as Rayleigh-Bénard convection after the first researchers to study the phenomenon experimentally [Bén01] and mathematically [Ray16]. The flow dynamics of fluids undergoing Rayleigh-Bénard convection are governed by the Rayleigh and Prandtl numbers. The Rayleigh number, Ra, is a dimensionless group defined as

$$\mathrm{Ra} = \frac{g\beta(T_H - T_C)H^3}{\nu\kappa}, \tag{1-1}$$

where g is the gravitational acceleration, β is the thermal expansion coefficient, T_H is the hotter temperature, T_C is the cooler temperature, H is the fluid height, ν is the kinematic viscosity, and κ is the thermal diffusivity. The Rayleigh number is the ratio of the buoyant forces to the inertial forces and can be physically viewed as a measure of the temperature gradient applied across the fluid. The flow regimes are also determined by the Prandtl number, defined as the ratio of the kinematic viscosity to the thermal diffusivity,

$$\Pr = \frac{\nu}{\kappa}. \tag{1-2}$$

The Prandtl number quantifies the ease to which the fluid will transport heat; the smaller the Prandtl number, the higher degree to which the fluid will transfer heat. In this study, the Prandtl number is assumed constant and can be considered characteristic of a particular fluid. For this study, liquid tin is the modeled fluid and its Prandtl number, Pr=0.008, is taken as a constant [Smi67]. Examples of fluids with "low" Prandtl numbers are molten metals and semiconductor melts – examples of "high" Prandtl numbers are water (Pr=6.9) and air (Pr=0.71).

The case of solutally-driven flow is simply a mass transfer analog of the above discussion. In such cases, the dynamic state of the fluid is a function of the solutal Rayleigh number defined by

$$\mathrm{Ra}_s = \frac{g\beta_c (C_H - C_C) H^3}{\nu D} \tag{1-3}$$

where, in this case, β_c is the solutal expansion coefficient, C is the concentration, and D is the mass diffusivity. The analogy of the Prandtl number in mass transfer is the Schmidt number defined by

$$\mathrm{Sc} = \frac{\nu}{D}. \tag{1-4}$$

Fluids undergoing Rayleigh-Bénard convection exhibit flow bifurcations at three different values of the Rayleigh number, denoted the first, second, and third critical Rayleigh numbers – Ra_{c1}, Ra_{c2}, and Ra_{c3}. If the fluid's Rayleigh number is less than Ra_{c1}, all heat transfer is through conduction, and is sufficient to maintain the buoyancy force below the magnitude necessary to overcome the system's inertia. As Ra_{c1} is surpassed, the fluid exhibits a cellular convective pattern that is steady with time and is geometrically dependent. Typical patterns developed in such flows are shown in Figure 1-2. For enclosures whose width and height are close to the same length (i.e., the aspect ratio is close to unity), a unicellular configuration is often observed. For tall enclosures (aspect ratio << 1), stacked cells tend to form, and for containers whose aspect ratios are >>1, torroids are commonly observed.

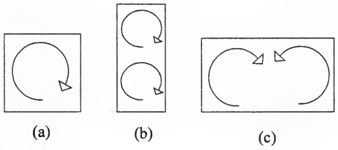


Figure 1-2 Typical flow patterns developed in Rayleigh-Bénard convection. (a) Unicellular ($\gamma \approx 1$); (b) Stacked Cells ($\gamma << 1$); and (c) Torroidal ($\gamma >> 1$).

If the Rayleigh number is further increased, beyond Ra_{c2}, the convection patterns begin to oscillate periodically with time. Finally, as Ra_{c3} is surpassed, the fluid motion becomes turbulent. The first critical Rayleigh number does not depend on the Prandtl number of the fluid (i.e., of the type of fluid that is studied) and has been determined previously by several authors [Cha61, Cat70, Cha70, Cha71, Cat72]. The second critical Rayleigh number, however, does depend on the Prandtl number, and the determination of

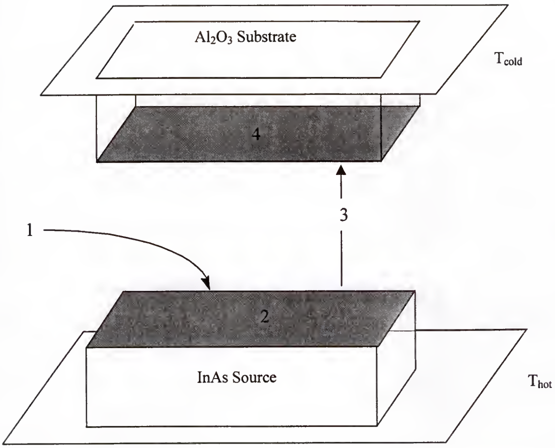
Ra_{c2} for liquid tin is the main focus of Chapter 5. The results are the first computations of the second critical Rayleigh numbers for fluids with Prandtl number as low as that of liquid tin. The transition to turbulence at Ra_{c3} and the resulting flow patterns is a very active area of research; however, their determination is beyond the scope of this work.

1.2 Crystal Growth from the Melt

Crystal growth from the melt generally has the distinct advantages of fast growth rate, relatively pure product, and a regular crystal structure [Roy92]. Moreover, experimental parameters of individual growth experiment are easy to adjust (*e.g.*, temperature). This is especially important when attempting to eliminate the adverse effects of convection on the crystal. If convection is detected in the melt, experimental conditions can be adjusted quickly to minimize the effect on the product. In light of this, this section discusses some popular crystal growth techniques for which convection plays an important role.

1.2.1 Closed Space Vapor Transport (CSVT)

Closed Space Vapor Transport (CSVT) is a crystal growth technique that has been observed to be influenced by convection. As shown in Figure 1-3 for the growth of InAs, CSVT is a film-growth method where a bulk material (the "source", *e.g.*, InAs) reacts with the gas ambient to form gaseous intermediaries (*e.g.*, InCl and As_4), which diffuse to a substrate (*e.g.*, Al_2O_3), and finally deposits a film (e.g., $InCl + \frac{1}{4}As_4 \rightarrow InAs\ (s) + HCl$) [Nic63]. A variety of materials have been grown with CSVT ranging from silicon [May65] and germanium [Tra69] to CdTe [Ant84] and $Zn_xCd_{1-x}Se$ [Ane96]. In CSVT, a source and substrate are held at different temperatures and thin film deposition is driven by the small shift of the chemical equilibrium produced by temperature



1. Inlet Gas Ambient
   2-3% $AsCl_3$ in H_2
   $2AsCl_3(v)+3H_2(v) \rightarrow 6HCl(v) + As_2(v)$
2. Surface Reaction
   $InAs(s) + HCl(v) \rightarrow InCl(v) + H_2(v)$
3. Transport of volatile species
   ($InCl$ and As_2) to substrate
4. Reverse of source reaction
   to form thin film
   $InCl(v) + H_2(v) \rightarrow InAs(s) + HCl(v)$

Figure 1-3. Schematic of Closed Space Vapor Transport (CSVT) technique.

differences. Although conducted with gaseous precursors, the temperature difference leads to the possibility of buoyancy-induced convection in much the same way as with melt crystal growth techniques. The adverse effect of convection on materials grown using CSVT has been experimentally observed for InAs films used in Hall device applications [Gri92]. This was the topic of some motivational research done for this project, the results of which are given in Appendix A.

1.2.2 Vertical Bridgman and Czochralski Techniques

Two of the most important crystal growth methods from melts are the Vertical Bridgman [Bri25] and Czochralski techniques [Czo18]. The Vertical Bridgman technique grows single crystals from a melt enclosed in a vertical ampoule with a series of heaters that impose an adjustable temperatures gradient along the ampoule. The upper zone is held at a higher temperature than the melting temperature of the crystal, while the lower zone is maintained below the melting temperature. Either moving the fluid in relation to the temperature gradient (the Bridgman method as shown in Figure 1-4) or moving the temperature in relation to the ampoule (freezing method) crystallizes the liquid. In the Czochralski technique, a crystal is pulled from the melt enclosed in a crucible, possibly rotating, by a single crystal seed as illustrated in Figure 1-5. Convection in such configurations can be induced in two ways: (1) thermal gradients encountered in these situations may lead to a cold-over-hot arrangement; and (2) over time, the segregation of impurities, dopants, or bulk material when growing alloys may produce a compositional variation. This can create a heavy-over-light solutal gradient and produce convection. Additionally, for Czochralski growth, thermal gradients may be induced through the lowering of the melt temperature that is needed for crystallization

[Kou96]. The Czochralski technique is a very popular method for growing group IV elemental semiconductors, such as silicon and germanium and many compound semiconductors (*e.g.*, GaAs, InP). In fact, nearly 95% of all single crystal silicon is grown using the Czochralski technique [Zul00].

The Czochralski technique is capable of producing very high purity crystals with very low defect concentrations needed for optical applications [Roy92]. Furthermore, it is accepted that convection is a significant experimental concern with both the Vertical Bridgman and Czochralski techniques. To eliminate the source of buoyancy, many researchers have advanced the idea of conducting melt crystal growth experiments in microgravity. In microgravity, the buoyancy induced by the density inversion is reduced, leaving the possibility of growing higher quality crystals from the melt. For example, a motivating study of this work was a microgravity study of $Pb_{1-x}Sn_xTe$ growth conducted by Matisak *et al.* [Mat97]. Such studies with alloys have proven to be extremely difficult on earth because of the significant segregation which occurs.

Unfortunately, microgravity crystal growth experiments still show evidence of convection-induced defects, induced most likely through: (1) surface tension gradients (i.e., Marangoni flow); and (2) gravity variations (i.e., gravity jitter). In microgravity, these effects are much more pronounced than on earth. There is, therefore, a need to develop techniques that characterize convection in a given experimental system operating under microgravity conditions. In this work, an electrochemical measurement technique is developed. This technique is discussed in more detail in the next sections.

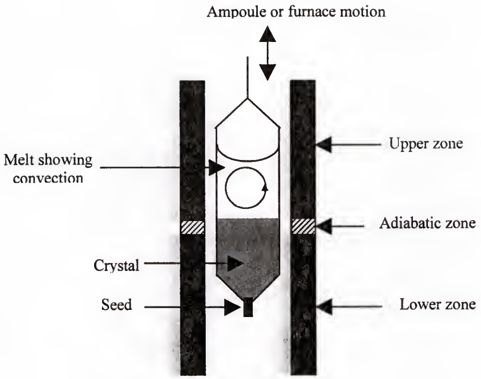


Figure 1-4. Vertical Bridgman crystal growth technique showing thermal gradients

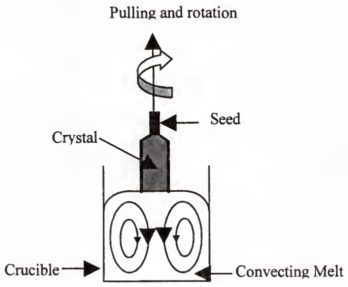


Figure 1-5. Czochralski crystal growth technique with rotation

1.3 Problem Statement: Flow Visualization of Low Prandtl Number Fluids

Because of the adverse effects of convection in crystal growth applications, researchers have developed several techniques of detecting convection in low Prandtl number fluids. Each of these visualization techniques has its own set of advantages and disadvantages which will be discussed in more detail in Chapter 6. At this point, though, it will suffice to say that most difficulties with flow visualization of low Prandtl number fluids arise for two reasons: (1) the opacity of the fluid; and (2) the high thermal conducting properties of the fluid. Because most low Prandtl-number fluids are opaque, tracer particles or dyes introduced into the system are not effective for visualizing flow streamlines (Even if they could, these would disturb the local flow characteristics and contaminate the crystal, thus reducing their usefulness). Also, the fact that the Prandtl number is low for a particular fluid must necessarily mean that the fluid transports heat rapidly; thus, temperature measurements are not useful. For this study, a novel technique to visualize convection in low Prandtl-number fluids has been developed that uses electrochemical sensors [Sea92a; Sea92b; Pra99].

1.4 Proposed Solution: Visualizing Flow Using Electrochemical Measurements

For the electrochemical technique developed in this study, the solid-state electrolyte, Yttria-Stabalized Zirconia (YSZ), is used. YSZ is useful because under specific electrochemical and oxygen partial pressure conditions, it can transport oxygen from a reference electrode into the liquid tin. As explained in more detail in Chapter 2, applying a potential across the YSZ produces an electrochemical reaction which forms oxygen ions. These ions transport through the YSZ by moving from vacancy to vacancy that are in the YSZ lattice. Finally, at the YSZ/tin interface, dissolved atomic oxygen in tin is formed. This oxygen packet is used as a tracer species. As the convecting fluid passes, the oxygen packet moves with the flow streamlines. In addition to being able to apply a given oxygen boundary condition, YSZ can also determine the oxygen concentration in the melt by measuring an open circuit EMF. As seen in Figure 1-6, the flow visualization system developed for this work has several YSZ sensors located along the periphery of the fluid container. As the oxygen tracer passes by, an open circuit EMF reading will measure the oxygen concentration. The convective pattern can be inferred by following the motion of the oxygen packet as it moves from sensor to sensor.

This technique has the advantage of avoiding the contamination problems of dyes and tracer particles. Furthermore, the concentration of oxygen present in the melt is several orders of magnitude lower that what is present in the melt natively, thus oxygen incorporation from the tracer species into the growing crystal is not an issue. The electrochemical measurement technique is also readily adaptable to microgravity studies because of its lightweight, its low power consumption, and its ability to be reinitialized.

For times less than the diffusion time (which is H^2/D, where H is the fluid height and D is the diffusivity), the diffusion of the oxygen tracer species can be neglected; however, it becomes important as the diffusive velocities exceeds the convective velocities. Thus, a complete characterization of the oxygen diffusivity in the model fluid (i.e., tin) is presented in this work.

1.5 Outline of This Work

1.5.1 Oxygen Concentration Cells

The theory of oxygen concentration cells is presented in Chapter 2. The method of determining oxygen concentration from EMF measurements is explained through the assumption of thermodynamic equilibrium across the electrochemical sensors. Finally, these results are combined with Fick's Law of diffusion to obtain the diffusivity in the liquid metal or alloy from EMF measurements

1.5.2 Computational Fluid Dynamics

In Chapter 3, the finite volume technique of solving the momentum and temperature/concentration balances is explained. This technique is used to obtain the critical Rayleigh numbers in Chapter 5. The equations of change are integrated across control volumes to represent the field variables (velocity, temperature, etc.) as a sum of the values at its nearest neighbors. Finally, its implementation in Cartesian and cylindrical coordinates is detailed.

1.5.3 Diffusivity of Oxygen in Liquid Tin and Tin-Lead Alloys

In Chapter 4, the measurement of the diffusivity of oxygen in liquid tin and tin-lead alloys is presented with emphasis on the temperature and concentration sensitivity of the diffusivity. In this chapter, evidence of convective effects in the melt is discovered.

1.5.4 Numerical Modeling of Rayleigh-Bénard Convection

In Chapter 5, flow bifurcations are characterized for a fluid with a very low Prandtl number, Pr=0.008 (liquid tin). These calculations emphasize the transition to oscillatory flow as a function of the container aspect ratio. Additionally, for lower aspect ratio enclosures, a mapping of the flow oscillations is constructed to visualize the effect of the oscillations on crystal growth experiments.

1.5.5 Detecting Convection Using Electrochemical Measurements

Natural convection is detected through electrochemical measurements in Chapter 6. Three geometries will be used – vertical fluid ampoules with concentration gradients, horizontal containers with concentration gradients, and vertical sensors with thermal gradients as shown in Figure 1-5. The operation of this prototype flow visualization sensor is presented for oscillating flows.

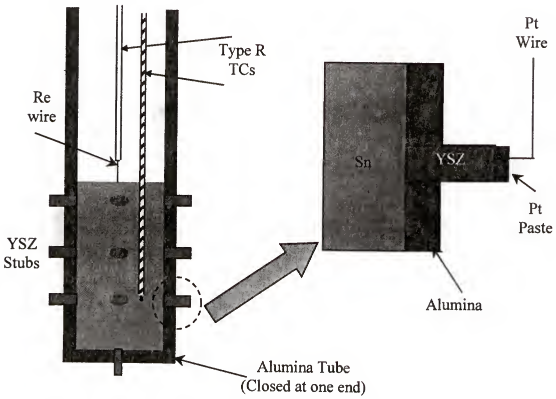


Figure 1-5 Schematic of electrochemical flow visualization apparatus developed in this work. Dissolved atomic oxygen is introduced into the melt through the bottom YSZ stub and follows the convective flow. The other YSZ sensors detect the passing oxygen.

CHAPTER 2
ELECTROCHEMICAL CELL THEORY

Coulometric titrations through solid electrolytes are the basis of the diffusivity and flow visualization experiments in this work. This is possible because of the oxygen transport capabilities of certain solid electrolytes, such as Yttria-Stabilized Zirconia (YSZ). In this chapter, background information about YSZ is presented, followed by a development of the important electrochemistry and thermodynamics behind the experiments presented in later chapters. The important mathematical relations governing oxygen transport are derived. This includes the Nernst equation for relating EMF to oxygen concentration and the flux law that relates current with concentration flux conditions. Finally, a similar derivation is presented for the case of an oscillating boundary condition problem to be used in future studies.

2.1 Operation of Yttria Stabilized Zirconia Electrolytes

When zirconium (IV) oxide ("zirconia", ZrO_2) is doped with various metal oxides, such as yttrium (III) oxide ("yttria", Y_2O_3), oxygen vacancies are formed to preserve electroneutrality between ZrO_2 and the trivalent Y_2O_3. Oxygen ions transport through easily through the YSZ via these vacancies at elevated temperatures. Most commercially available YSZ is doped between 8 and10 mole percent, which is the minimum that preserves the original fluorite structure of ZrO_2. All experiments in this work are performed with 8 mole % YSZ.

2.1.1 Electrolytic Domain

Electrochemical sensors using YSZ must be operated in the "electrolytic domain," a regime defined as that in which 99% of all current transporting through the material is through ionic conduction. This is determined by the ionic transference number, t_{ion}, defined as

$$t_{ion} = \frac{\sigma_{ion}}{\sigma_{ion} + \sigma_n + \sigma_p}, \qquad (2\text{-}1)$$

where σ is the conductivity. The ionic transference number is simply the fraction of current carried by ions (σ_{ion}) compared with that for electrons (σ_n) and holes (σ_p). The electrolytic domain was characterized by [Kle82] for 9 mole percent YSZ and shows the temperature and oxygen partial pressure ranges over which YSZ behaves as an electrolyte (Figure 2-1). The electrolytic domain for the 8 mole percent YSZ used in this study is assumed not to deviate from this observation. The oxygen partial pressure for applications studied in this work ranges between 10^{-21} atm to 0.21 atm and for temperature between 600 °C and 800 °C, conditions well within the electrolytic domain. As a rule of thumb, YSZ can usually be used as a solid state electrolyte in the temperature range of 600 °C to 2000 °C.

2.1.2 Response Time

The response time of YSZ sensors was studied by [Win85], who reports that YSZ response times are of the order of a few milliseconds for the temperature range investigated in this work. As computed in Chapter 4, the convective velocities of the fluids in this study are of the order of 1-2 cm/s. The spacing between YSZ sensors is of the order of 1 cm, leading to a travel time between YSZ sensors of the order of 1 s; therefore, the YSZ sensors are well-suited for flow-sensing applications.

2.2 Thermodynamic Analysis

The thermodynamic background is now be presented that will relate the measured EMF to oxygen concentration that is needed to interpret the diffusivity and flow visualization experiments. In a titration experiment, oxygen is transported in ionic form from a reference electrode, through the electrolyte, and dissolves in atomic form into the working electrode, as schematically depicted in Figure 2-2. Figure 2-2 shows the reference electrode for diffusion studies, a 50 mole% mixture of powdered Cu/Cu_2O; the reference electrode for the flow visualization experiments is air (The oxygen titration process explained in this chapter is the same regardless of the reference electrode).

Potential measurements can be made across the YSZ and related to the oxygen concentration assuming equilibrium at the electrode/YSZ interface. The condition for thermodynamic equilibrium is that the electrochemical potential across the cell is constant. The electrochemical potential is defined as

$$\eta = \mu + nF\phi, \tag{2-2}$$

where η is the electrochemical potential, μ is the chemical potential of oxygen, F is Faraday's constant (=96,487 C/eq), and ϕ is the electrical potential. The stability criterion requires that

$$\eta_1 = \mu_1 + nF\phi_1 = \eta_2 = \mu_2 + nF\phi_2, \tag{2-3}$$

where 1 and 2 indicates either side of the YSZ. The chemical potential can be written as

$$\mu = \mu_o + RT\ln(a), \tag{2-4}$$

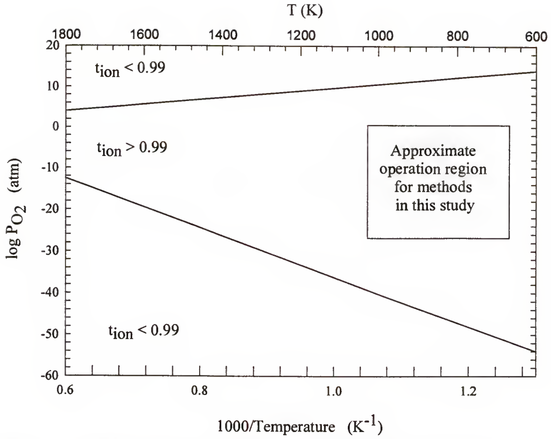
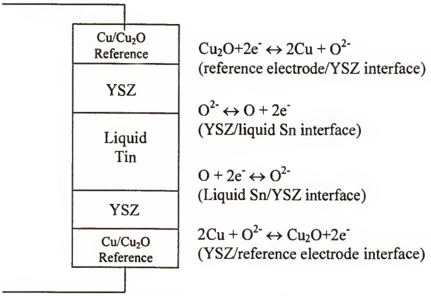


Figure 2-1. The electrolytic domain of 9% Yttria Stabilized Zirconia [Kle82].



Cu/Cu_2O Reference
YSZ
Liquid Tin
YSZ
Cu/Cu_2O Reference
$Cu_2O+2e^- \leftrightarrow 2Cu + O^{2-}$
(reference electrode/YSZ interface)
$O^{2-} \leftrightarrow O + 2e^-$
(YSZ/liquid Sn interface)
$O + 2e^- \leftrightarrow O^{2-}$
(Liquid Sn/YSZ interface)
$2Cu + O^{2-} \leftrightarrow Cu_2O+2e^-$
(YSZ/reference electrode interface)

Figure 2-2. Diffusivity measurements showing reactions at the different interfaces. (For flow visualization studies, air is the reference electrode).

where a is the activity of oxygen at the appropriate location. For sufficiently dilute solutions, Henry's Law is assumed, rendering the activity as a = γc, where γ is the activity coefficient (assumed to be unity) and c is the concentration. Substitution into Equation (2-3) yields

$$-RT\ln\left(\frac{C_2}{C_1}\right) = nF(\phi_2 - \phi_1) \quad . \tag{2-5}$$

The potential difference, ϕ_2-ϕ_1 is equivalent to the EMF difference, E_2-E_1, so

$$-RT\ln\left(\frac{C_2}{C_1}\right) = nF(E_2 - E_1) \quad , \tag{2-6}$$

or

$$C_2 = C_1 \exp\left[-\frac{nF}{RT}(E_2 - E_1)\right], \tag{2-7}$$

which is called the Nernst Equation. A few comments can now be made about the Nernst Equation:

- Equation (2-7) states very compactly that the EMF is simply another representation of the oxygen concentration at a specified point in the fluid given the concentration of the reference electrode (C_1). Therefore, a measurement of the EMF gives the oxygen concentration.
- The derivation so far has assumed nothing about the potential field ϕ_2-ϕ_1 (or, E_2-E_1). Thus, the potential may be either applied or measured (i.e., applied or measured concentration).
- The Nernst equation applies to any two EMF/concentration values. In other words, if the concentration is known for any EMF value, all other concentrations can be calculated knowing only the measured potential. This is useful because both the saturation concentration and potential have been well characterized for tin and are used as C_1 and E_1 throughout this work. The relation between EMF and concentration using the saturation conditions as a basis is shown in Figure 2-3.

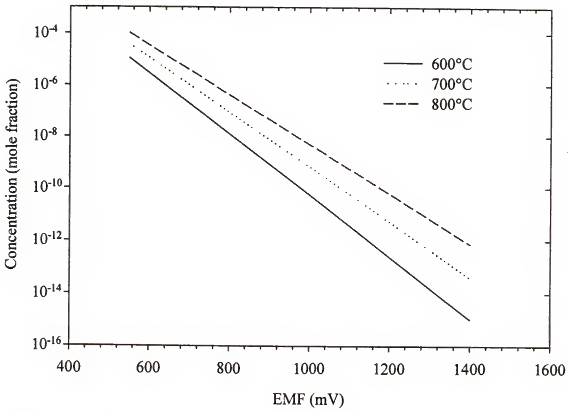


Figure 2-3. Oxygen concentrations corresponding to various EMF values as a function of temperature using oxygen saturation as basis condition.

2.3 Diffusivity Measurements

In the diffusivity studies, the reference electrode is a 50 mole % mixture of Cu/Cu_2O. The half reactions are

$$Cu_2O(pure) + 2e^-(Cu) \leftrightarrow 2Cu(pure) + O \quad (2\text{-}8)$$

and

$$O^{-2}(YSZ) \leftrightarrow [O]_{Sn} + 2e^-(Sn) \ , \quad (2\text{-}9)$$

where $2e^-(Cu)$ indicates two free electrons in copper, and $[O]_{Sn}$ is the oxygen dissolved in the liquid metal or alloy. The overall reactions are, chemically,

$$Cu_2O(pure) \leftrightarrow 2Cu(pure) + [O]_{Sn} \ , \quad (2\text{-}10)$$

and electrically as,

$$2e^{-}(\mathrm{Cu}) \rightarrow 2e^{-}(\mathrm{Sn})\,. \tag{2-11}$$

The oxygen diffusivity in liquid metals can be measured in a variety of cell configurations and the governing equations will now be presented.

2.3.1 Transient EMF Measurements

Transient measurements of the diffusivity of oxygen in a liquid metal can be made either through a depletion or introduction experiment ("pump out" or "pump in"). In each of these experiments, an initial concentration of oxygen is established throughout the melt by applying a known voltage and allowing the system to equilibrate. For a depletion experiment, the initial oxygen concentration is relatively high and at the initiation of the experiment, oxygen is depleted from either the top or bottom by applying a large potential to the cell, corresponding to an oxygen concentration of nearly zero. The EMF response at the opposite end of the melt is then tracked with time. To relate these EMF measurements to the diffusivity, Fick's Law must be solved:

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial z^2}, \tag{2-12}$$

where the following has been assumed: (1) diffusion is one-dimensional in the axial direction; (2) convection is absent; and (3) the diffusivity is not position-dependant. For a depletion experiment, the driving potential is assumed to impose a zero boundary condition (it is actually about 1 part in 10^{12}). The other boundary conditions are

$$\begin{array}{ll} c(t=0)=c_O & \text{All } z \\ c(t>0)=0 & z=0 \\ \dfrac{\partial c}{\partial z}(t\geq 0)=0 & z=H \end{array} \tag{2-13}$$

The solution to this system is provided in Appendix B, and the result is

$$\frac{c(z,t)}{c_o}=\sum_{n=0}^{\infty}\frac{\exp\left[\frac{-D}{H^2}\left(n+\frac{1}{2}\right)^2\pi^2 t\right]}{\left(n+\frac{1}{2}\right)\frac{\pi}{2}}\sin\left(n+\frac{1}{2}\right)\pi\frac{z}{H}, \tag{2-14}$$

where c_o is the initial concentration and H is the height of the fluid. The oxygen concentration at the edge of the system, z=H, (where the sensing EMF measurement is made) is

$$\frac{c(H,t)}{c_o}=\sum_{n=0}^{\infty}\frac{2(-1)^n}{\left(n+\frac{1}{2}\right)\pi}\exp\left[-\frac{D}{H^2}\left(n+\frac{1}{2}\right)^2\pi^2 t\right]. \tag{2-15}$$

At larger times, only the first term in the infinite series is important and the equation becomes

$$\frac{c(H,t)}{c_o}=\frac{4}{\pi}\exp\left[-\left(\frac{D}{H^2}\right)\left(\frac{\pi^2}{4}\right)t\right]. \tag{2-16}$$

Combining (2-16) and the Nernst equation gives

$$\frac{2F(E(t)-E_o)}{RT}=\left(\frac{\pi^2}{4}\frac{D}{H^2}\right)t-\ln\left(\frac{4}{\pi}\right). \tag{2-17}$$

From this equation, it is seen that E(t)-E_o is linear function of time with a slope of $\frac{\pi^2}{4}\frac{D}{H^2}\frac{RT}{2F}$ and intercept of $-\frac{RT}{2F}\ln\left(\frac{4}{\pi}\right)$. The diffusivity can, therefore, be extracted by analyzing the slope of the long-time and linear portion of the E_t-E_o vs. time curve. Moreover, the intercept of Equation (2-17) is independent of diffusivity, and as such, can be used as an independent measure of the quality of a particular result.

The development is slightly different when an introduction ("pump in") experiment is performed. The boundary conditions are now

$$\begin{aligned} c(t=0) &= c_o \\ c(t>0) &= C_{bottom} \\ \frac{\partial c}{\partial z}(t \geq 0) &= 0 \end{aligned} \qquad (2\text{-}18)$$

,

where c_{bottom} indicates a new, non-zero, boundary condition. The treatment of equations (2-14) through (2-16) can be used by transforming the concentration, c, by $c^{*}=c-c_{bottom}$. All boundary conditions are now zero and the diffusion equation, (2-12), can be solved in the same way as in Appendix B. The result corresponding to Equation (2-16) is

$$\frac{c^{*}(H,t)}{c_o^{*}} = \frac{4}{\pi}\exp\left[-\left(\frac{D}{H^2}\right)\left(\frac{\pi^2}{4}\right)t\right] \qquad (2\text{-}19)$$

At this point, the diffusivity can be related to the slope of the curve of $\ln\left(\frac{C^{*}}{C_o^{*}}\right)$ versus time.

2.3.2 Current Measurements

To this point, only EMF measurements have been related to the diffusivity. The diffusivity can also be determined via current measurements under either steady-state or transient operation. In either case, current and diffusivity can be related through the electroneutrality condition,

$$D\left|\frac{\partial c}{\partial z}\right| = \frac{I}{nFA}, \qquad (2\text{-}20)$$

where D is the diffusivity, dc/dz is the concentration gradient, I is the current, n is the electrochemical valency, F is the Faraday constant (96,487 C/eq), and A is the cross-sectional area of the medium through which diffusion occurs. Thus, EMF is related to oxygen concentration, while the current is related to the flux of oxygen through the YSZ.

In this work, both steady and transient current measurements will be used. For steady-state current experiments, a concentration gradient is applied across the fluid and the resulting currents are related to the diffusivity through Equation (2-20).

Transient current measurements can also be made. For such experiments, the concentration gradient is obtained by differentiating Equation (2-15), and substituting the result into Equation (2-20) resulting in

$$\ln(I) = \left(\frac{4C_0DFA}{H}\right) - \left(\frac{D\pi^2}{4H^2}\right)t\,. \tag{2-22}$$

The dynamic response of the current can now be related to the diffusivity. Unlike the EMF measurement analysis in Equation (2-17), both the slope and intercept of Equation (2-22) are diffusivity-dependant.

2.3.3 Flow Visualization Experiments

Convection visualization experiments consist of a working and reference electrode separated by a YSZ electrolyte. The working electrode is oxygen dissolved in liquid tin and the reference electrode is air. This system is represented as

$$\mathrm{Pt} \mid \mathrm{Re} \mid [\mathrm{O}]_{\mathrm{Sn}} \parallel \mathrm{YSZ} \parallel \mathrm{O_2(g)} \mid \mathrm{Pt}, \tag{2-23}$$

where $[O]_{Sn}$ indicates oxygen dissolved in liquid tin, and Re is rhenium wire which forms a contact between the liquid tin and the platinum wire. The reactions taking place as the oxygen becomes dissolved are

$$\begin{aligned} &\tfrac{1}{2}O_2(g) + 2e^- \leftrightarrow O^{2-} \\ &O^{2-} \leftrightarrow [O]_{Sn} + 2e^- \end{aligned} \tag{2-24}$$

The overall reaction for this process is simply:

$$\tfrac{1}{2}O_2(g) \rightarrow [O]_{Sn}, \qquad (2\text{-}25)$$

that is, oxygen gas from the ambient is transformed into dissolved oxygen. The form of the Nernst equation that is important here is

$$E_t - E_0 = \frac{RT}{nF}\ln\left(\frac{C_t}{C_0}\right), \qquad (2\text{-}26)$$

or solving for concentration yields

$$C = C_0 e^{\left[\frac{-nF(E-E_0)}{RT}\right]}. \qquad (2\text{-}27)$$

Therefore, the oxygen concentration at any YSZ sensor can be determined by measuring the voltage at the sensor during the course of an experiment.

2.4 Periodic Boundary Conditions

Diffusivities can also be measured through application of periodic concentration boundary conditions. Simultaneous current measurements can be made and related to the diffusivity. With this in mind, equations similar to equations (2-22) will be derived for this configuration [Bir02]. Note that the argument below is valid only for a constant applied frequency.

The physical system is identical to that with the steady boundary except: (1) the geometry is taken to be semi-infinite; and (2) the following boundary conditions are used:

$$c(z=0,t) = c_{amp}R\left\{e^{i\omega t}\right\} \qquad (2\text{-}28)$$

$$c(z \rightarrow \infty, t) \rightarrow c_{init} \qquad (2\text{-}29)$$

where c_{amp} indicates the amplitude of the periodic concentration boundary condition and R{-} indicates the real part of the argument. The one-dimension diffusion equation is used as before:

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial z^2}\,. \tag{2-30}$$

To make the boundary condition homogeneous, the substitution $\hat{c} = c - c_{\text{init}}$ is made as in Equation (2-18), ensuring that the time and space derivatives of $\hat{c}$ are the same as for c ($\hat{c}$ will be redefined as c from this point on). The solution for the system (2-28) through (2-30) is postulated to be of the form

$$c = R\{\tilde{c}(z)e^{i\omega t}\}, \tag{2-31}$$

where $\tilde{c}(z)$ is a complex function of position only. Substituting this trial solution into (2-30) gives

$$\frac{\partial}{\partial t}\left[R\{\tilde{c}e^{i\omega t}\}\right] = D\frac{\partial^2}{\partial z^2}\left[R\{\tilde{c}e^{i\omega t}\}\right]; \tag{2-32}$$

simplifying gives,

$$R\{\tilde{c}i\omega e^{i\omega t}\} = R\left\{D\frac{\partial^2\tilde{c}}{\partial z^2}e^{i\omega t}\right\}. \tag{2-33}$$

Equating the arguments of Equation (2-33) and canceling the exponential terms results in the following ODE:

$$\frac{d^2\tilde{c}}{dz^2} - \left(\frac{i\omega}{D}\right)\tilde{c} = 0\,; \tag{2-34}$$

the boundary conditions then become

$$\tilde{c}(z=0) = c_{\text{amp}} \tag{2-35}$$

$$\tilde{c}(z\to\infty)\to 0\,. \tag{2-36}$$

The solution to (2-34) is

$$\tilde{c} = C_1 e^{\sqrt{\frac{i\omega}{D}} z} + C_2 e^{-\sqrt{\frac{i\omega}{D}} z}. \tag{2-37}$$

Equation (2-37) can be simplified by noting that

$$\sqrt{i} = \pm\frac{1+i}{\sqrt{2}}, \tag{2-38}$$

resulting in

$$\tilde{c} = C_1 e^{\sqrt{\frac{\omega}{2D}}(1+i)z} + C_2 e^{-\sqrt{\frac{\omega}{2D}}(1+i)z}. \tag{2-39}$$

Boundary condition (2-36) requires C_1=0, while by inspection, boundary condition (2-35) gives C_2=c_{amp}. The solution is then given by

$$\tilde{c} = c_{amp} e^{-\sqrt{\frac{\omega}{2D}}(1+i)z}. \tag{2-40}$$

Substituting (2-40) into (2-31) gives the final solution as

$$c = R\left\{ c_{amp} e^{-\sqrt{\frac{\omega}{2D}}(1+i)z} e^{i\omega t} \right\} = c_{amp} e^{-\sqrt{\frac{\omega}{2D}} z} R\left\{e^{i\omega t}\right\}, \tag{2-41}$$

which is equivalent to

$$c = c_{amp} e^{-\sqrt{\frac{\omega}{2D}} z} \cos\left(\omega t - \sqrt{\frac{\omega}{2D}} z\right) - c_{init}, \tag{2-42}$$

where c_{init} resurfaces to account for the non-homogenous boundary condition. To relate this to current measurements, the spatial derivative of this quantity is needed which is

$$\frac{dc}{dz} = -c_{amp} e^{\sqrt{\frac{\omega}{2D}}} \sqrt{\frac{\omega}{2D}} \left[\cos(\omega t) - \sin(\omega t)\right]. \tag{2-43}$$

If current measurements are desired on the face the enclosure, then z=0, and the current is obtained through substitution into Equation (2-20) which yields

$$I = -D(nFA)c_{amp}e^{\sqrt{\frac{\omega}{2D}}}\sqrt{\frac{\omega}{2D}}\left[\cos(\omega t) - \sin(\omega t)\right]. \qquad (2\text{-}44)$$

The solution to Equation (2-44) for various diffusivities and a fixed frequency of 1 Hz is shown in Figure 2-4 showing that the diffusivity effects the amplitude of the (periodic) measured current but not the frequency or phase shift relative to the forcing potential. The oxygen diffusivity, then, can be determined through measuring the amplitude of the current response that is measured. In Figure 2-5, the current response for various concentration frequencies is given for a fixed diffusivity of 10^{-5} cm^2/s. Figure 2-4 shows that the response depends greatly on the applied concentration frequency.

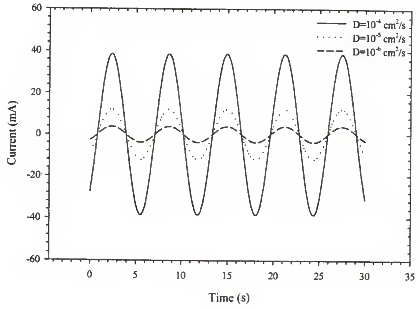


Figure 2-4. Current response for various diffusivities and a fixed concentration frequency of 1 Hz. A=25 mm^2, c_{amp}=10^{-6} mole fraction.

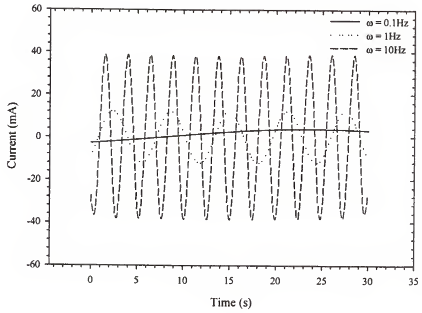


Figure 2-5 Current response for various frequencies and a fixed diffusivity of 10^{-5} cm^2/s, A=25 mm^2, c_{amp}=10^{-6} mole fraction.

CHAPTER 3
COMPUTATIONAL FLUID DYNAMICS USING THE *SIMPLE* ALGORITHM

In this work, computational fluid dynamics simulations are used to predict the velocity and temperature profiles resulting from the heating of low Prandtl-number fluids from below. In this chapter, the theory used in developing these numerical models is detailed. For this work, a control volume algorithm is used for all calculations. In this approach, the equations of change are integrated over a small control volume and the field variables (velocity, temperature, and concentration) are converged iteratively. Control volume approaches are advantageous over similar techniques, such as finite-element analysis, for the following reasons:

- The technique normally takes less computer time than other techniques to produce converged solutions.
- The codes are relatively straightforward to develop.
- The converged fields are guaranteed to satisfy the integral conservation equations over any grid spacing.

Such algorithms also have the following disadvantages:

- They are difficult to adapt to multicomponent systems or irregular-shaped geometries because of the rectangular grids that must be used.
- They require staggered grids to be used for the different field variables, increasing the difficulty of debugging.

Specifically, this work uses the SIMPLE algorithm (Semi-Implicit Method of Pressure Linking Equations) of Patankar [Pat80]. The assumptions made for this procedure will be explained in due course; however, some introductory concepts must first be explained.

3.1 Derivation of the Equations of Change

Before proceeding, a brief discussion of the equations of change would be helpful. For a more thorough discussion, the reader is referred to well-known textbooks in transport phenomena [Bir60, Fah83].

3.1.1 Equation of Continuity

The equation of continuity is simply a total mass balance over a stationary control volume. The balance is expressed as

$$\begin{matrix}\text{Rate of Mass} \\ \text{Accumulation}\end{matrix} = \begin{matrix}\text{Rate of Mass} \\ \text{In}\end{matrix} - \begin{matrix}\text{Rate of Mass} \\ \text{Out}\end{matrix}$$

This can be expressed mathematically on a volume basis as

$$\Delta x \Delta y \Delta z \frac{\partial \rho}{\partial t} = \Delta y \Delta z\left[\left(\rho v_x\right)\big|_x - \left(\rho v_x\right)\big|_{x+\Delta x}\right] + \Delta x \Delta z\left[\left(\rho v_y\right)\big|_y - \left(\rho v_y\right)\big|_{y+\Delta y}\right] + \ldots$$
$$\ldots + \Delta x \Delta y\left[\left(\rho v_z\right)\big|_z - \left(\rho v_z\right)\big|_{z+\Delta z}\right] \tag{3-1}$$

where ρ is the density, v_x, v_y, and v_z are the velocity components, and x, y, and z are the Cartesian directions. If this expression is divided by $\Delta x \Delta y \Delta z$ and the limit is taken as x,y, and z approach zero, the following is obtained

$$\frac{\partial \rho}{\partial t} = -\left(\frac{\partial(\rho v_x)}{\partial x} + \frac{\partial(\rho v_y)}{\partial y} + \frac{\partial(\rho v_z)}{\partial z}\right). \tag{3-2}$$

This can also be represented vectorally as

$$\frac{\partial \rho}{\partial t} + \nabla \bullet (\rho \vec{v}) = 0. \tag{3-3}$$

Equation (3-3) is referred to as the Equation of Continuity. In Chapter 5, it is proven that the Continuity equation for natural convection systems encountered in this work can be simplified to the form

$$\nabla \bullet \vec{v} = 0. \tag{3-4}$$

3.1.2 Conservation of Momentum

A similar treatment can be used to obtain velocity profiles through a linear momentum conservation argument. The balance argument can be physically justified as a balance:

$$\begin{matrix}\text{Rate of Momentum} \\ \text{Accumulation}\end{matrix} = \begin{matrix}\text{Flux from} \\ \text{Conduction}\end{matrix} + \begin{matrix}\text{Flux from} \\ \text{Convection}\end{matrix} + \begin{matrix}\text{Flux from} \\ \text{Surface Forces}\end{matrix} + \begin{matrix}\text{Flux from} \\ \text{Body Forces}\end{matrix} \qquad (3\text{-}5)$$

The terms in Equation (3-5) can be represented as follows:

Accumulation Term: $\frac{\partial}{\partial t}(\rho\vec{v})$

Conduction Term: $\mu\nabla^2\vec{v}$

Convection Term: $-\rho\vec{v}\bullet\nabla(\rho\vec{v})$

Surface Force Term: $-\nabla p$

Body Force Term: $\rho\vec{g}$

where: μ is the dynamic viscosity

p is the thermodynamic pressure

$\vec{g}$ is the gravitational vector

The final form of the linear momentum balance that is used in this work is termed the Navier-Stokes equation and is represented by

$$\frac{\partial(\rho\vec{v})}{\partial t} + \rho\vec{v}\bullet\nabla(\rho\vec{v}) = -\nabla p + \mu\nabla^2\vec{v} + \rho\vec{g}\,. \qquad (3\text{-}6)$$

3.1.3 Conservation of Energy

A similar statement of the conservation of energy can also be written by inspection as

$$\frac{\partial(\rho C_p T)}{\partial t} + \rho\vec{v} \bullet \nabla(\rho C_p T) = k\nabla^2(\rho C_p T) \qquad (3\text{-}7)$$

where: k is the thermal conductivity

C_p is the specific heat; and

T is the absolute temperature.

The balance equations (3-4), (3-6), and (3-7) are unsolvable analytically; therefore, to obtain velocity and/or temperature fields, one must do one or more of the following:

Make simplifying assumptions. This is usually done by limiting the dimensionality of the problem or neglecting the contribution of one or more terms in the balance equation relative to others.
Find steady state solutions and testing the stability of the result to perturbations. This is the well-known linear stability technique and has proven effective in analyzing many natural convection problems, but can only give steady state solutions.
Solve the equation of change numerically. This separates the computational domain into several discrete elements and solves the equations of change over each control volume. This is the topic of the rest of this chapter.

3.2 Solution Procedure

In control volume approaches, the equations of change are transformed into sums of nearest neighbor interactions. For example, a field variable ϕ at any point P, ϕ_P, is rendered as

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + b$$

where ϕ_E, ϕ_W, ϕ_N, and ϕ_S indicates the field variable on the grid point to the "east", "west", "north", and "south" of ϕ_P. The term "b" is called the mass source term and accounts for the value of ϕ_P at the previous time step. By examining the coefficients a_E...a_S, one can make a very important conclusion that will be helpful in the rest of this presentation. Since the value of the field variable depends on the values of its nearest neighbors, an increase in ϕ at one point leads to an increased ϕ at its neighboring grid points. Stated differently, it is physically impossible for an increase in ϕ to lead to a decrease in ϕ at an adjacent point. This leads to an extremely important principle that will be called the Positive Coefficients Principle. It is stated compactly as:

> The values of the coefficients of the discretized conservation equations must all be either positive or zero.

If the field variable is temperature, for example, this principle is simply a statement of the Second Law of Thermodynamics – heat travels from hot to cold.

3.3 Discretization of the Equations of Change

To solve for continuous variables at specified grid points, the computational domain and the equations of change must be discretized. The method of doing this is illustrated best by an example. At this point, consider a one-dimensional heat conduction problem in Cartesian co-ordinates (i.e., the one-dimensional analogue of Equation (3-7), neglecting the convection and dissipation terms). If the density and specific heat are taken to be constants, the expression for (3-7) is

$$\rho C_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) . \tag{3-8}$$

Integrating this in space and time obtains

$$\rho C_p \int_w^e \int_t^{t+\Delta t} \frac{\partial T}{\partial t} dtdx = \int_t^{t+\Delta t} \int_w^e \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) dxdt , \tag{3-9}$$

where "e" and "w" in the integration limits indicate "east" and "west" referring to cartographic direction in which the integration is performed. For the LHS of Equation (3-9), we assume that the value of the temperature is constant throughout the control volume. The LHS is then

$$\rho C_P \int_w^e \int_t^{t+\Delta t} \frac{\partial T}{\partial t} dt dx = \rho C_P \Delta x \left(T_P^1 - T_P^0 \right). \tag{3-10}$$

To obtain the RHS, a linear temperature profile is assumed throughout the control volume (called the "piecewise-linear profile"). This results in

$$\rho C_P \Delta x \left(T_P^1 - T_P^0 \right) = \int_t^{t+\Delta t} \frac{k_e (T_E - T_P)}{(\delta x)_e} - \frac{k_w (T_P - T_W)}{(\delta x)_w} dt. \tag{3-11}$$

Note that the subscript "E" refers to the value at the <u>point</u> east of p and "e" refers to the value at the eastern end of the <u>control volume</u>. To determine the effect of time on the temperature, it was proposed by Patankar [Pat80] to use a weighting factor, f, *viz.*,

$$\int_t^{t+\Delta t} T_P dt = \left[f T_P^1 + (1-f) T_P^0 \right] \Delta t. \tag{3-12}$$

Similar expressions can be written for T_E and T_W. If this weighting factor expression is substituted into Equation (3-11), the following is obtained

$$\rho C_P \frac{\Delta x}{\Delta t} \left(T_P^1 - T_P^0 \right) = f \left[\frac{k_E \left(T_E^1 - T_P^1 \right)}{(\delta x)_E} - \frac{k_W \left(T_P^1 - T_W^1 \right)}{(\delta x)_W} \right] + \ldots$$
$$\ldots + (1-f) \left[\frac{k_E \left(T_E^0 - T_P^0 \right)}{(\delta x)_E} - \frac{k_W \left(T_P^0 - T_W^0 \right)}{(\delta x)_W} \right]. \tag{3-13}$$

At this point, redefine the temperature at time t+Δt, T^1, as T. Then,

$$a_P T_P = a_E \left[f T_E + (1-f) T_E^0 \right] + a_W \left[f T_W + (1-f) T_W^0 \right] + \left[a_P^0 - (1-f) a_E - (1-f) a_W \right] T_P^0. \tag{3-14}$$

where: $a_E = \dfrac{k_E}{(\delta x)_E}$

$$a_W = \frac{k_W}{(\delta x)_W}$$

$$a_P^0 = \frac{\rho C_p \Delta x}{\Delta t}, \text{ and}$$

$$a_P = f a_E + f a_W + a_P^0, \text{ and}$$

$$b = \left[a_P^0 - (1-f)a_E - (1-f)a_W\right]T_P^0$$

3.3.1 Time-Stepping Schemes

There are three important schemes that are used to determine the value of the weighting factor, f. They are now explained in more detail.

3.3.1.1 Explicit

The explicit scheme is when f is set to 0. If this scheme is used, Equation (3-14) becomes

$$a_P T_P = a_E T_E^0 + a_W T_W^0 + \left(a_P^0 - a_E - a_W\right)T_P^0. \qquad (3\text{-}15)$$

This procedure is termed the "explicit scheme" because the temperature at any point is dependant only upon the field variable at previous time steps and not upon its current value. As a result, an equation similar to (3-15) must be written for T_W and T_E, resulting in tripling in computational time from other assumptions. It can be seen, therefore, that the apparent simplicity of the explicit time marching scheme actually complicates the numerical procedure significantly.

3.3.1.2 Crank-Nicholson

The Crank-Nicholson scheme is when f=0.5 and seeks to account for the fact that the fluid history can influence the present dynamic state. If this is used, the b term in

Equation (3-14) becomes $a_P^0 - \frac{1}{2}(a_E + a_W)$. If a uniform grid spacing is used, this is equivalent to

$$\rho C_p \frac{\Delta x}{\Delta t} - \frac{k}{\Delta x}. \qquad (3\text{-}16)$$

From this expression, it can be seen that if Δt is too large or Δx is too small, this term can be negative. If this is true, then an increase in temperature can cause a decrease in temperature at its nearest neighbors, which violates the Positive Coefficients Principle. In natural convection phenomena, effects on very small length scales influence flow dynamics; thus, simulations must be performed over small integration lengths. It is for this reason that the Crank-Nicholson scheme is not used in this work.

3.3.1.3 Fully Implicit

The fully implicit time scheme is used in this work and requires f=1. Its selection arises from its guarantee that all nearest-neighbor coefficients are positive. When this is used, Equation (3-14) is transformed into

$$a_P T_P = a_E T_E + a_W T_W + b \qquad (3\text{-}17)$$

where: $a_E = \frac{k_E}{(\delta x)_E}$

$$a_W = \frac{k_W}{(\delta x)_W}$$

$$a_P^0 = \frac{\rho C_p \Delta x}{\Delta t}$$

$$b = a_P^0 T_P^0$$

$$a_P = a_E + a_W + a_P^0.$$

A similar equation can be developed for higher dimensional problems by adding additional nearest neighbor coefficients (named “up”, “down”, “top”, and “bottom”).

3.3.2 Addition of Convection

If convective effects are included in the field analysis, one of several computational schemes can be used which will be explained later in this section. For the moment, consider the one-dimensional conduction and convection problem,

$$\frac{d}{dx}(\rho u\phi)=\frac{d}{dx}\left(\Gamma\frac{d\phi}{dx}\right) \tag{3-18}$$

where: ϕ is the field variable (velocity or temperature)

Γ is a “diffusion” coefficient (for velocity calculations, this is the kinematic viscosity; for temperature profiles, this is the thermal conductivity).

If Equation (3-18) is integrated over a one-dimensional control volume, the following is obtained:

$$(\rho u\phi)_e-(\rho u\phi)_w=\left(\Gamma\frac{d\phi}{dx}\right)_e-\left(\Gamma\frac{d\phi}{dx}\right)_w . \tag{3-19}$$

The question is now how to represent the derivatives of the field variable, ϕ, in the derivatives of the convection terms. There are several techniques that are used to do this, which are now explained.

3.3.2.1 Central-Difference

This procedure is the analogue of the piecewise-linear assumption used for the conduction analysis. This represents the field variable as

$$\phi_E=\frac{1}{2}(\phi_e+\phi_P) \tag{3-20}$$

and

$$\phi_W = \frac{1}{2}(\phi_P + \phi_W), \tag{3-21}$$

where, again, the lower-case subscripts indicate variable value at the control volume interface and the upper-case indicates the value at the grid point. Equation (3-19) is then transformed into

$$\frac{1}{2}(\rho u)_e(\phi_e + \phi_P) - \frac{1}{2}(\rho u)_w(\phi_P + \phi_W) = \frac{\Gamma_E(\phi_e - \phi_P)}{(\delta x)_e} - \frac{\Gamma_W(\phi_P - \phi_W)}{(\delta x)_W}. \tag{3-22}$$

The following definitions can now be made: $F \equiv \rho u$ and $D \equiv \frac{\Gamma}{\delta x}$, where F is a convective flux and D is a diffusive flux. Equation (3-22) is now transformed into a nearest-neighbor form as

$$a_P\phi_P = a_E\phi_e + a_W\phi_W \tag{3-23}$$

where: $a_E = D_E\phi_e - \frac{F_E}{2}$ (3-24)

$$a_W = D_W\phi_W + \frac{F_W}{2} \tag{3-25}$$

$$a_P = a_E + a_W + (F_e - F_W) \tag{3-26}$$

The problems associated with the central-difference scheme are analogous to those of the piecewise-linear discretization method as seen in equations (3-24) and (3-25). In these equations, the coefficients a_w and a_E can become negative when $|F| > 2D$, thus violating the Positive Coefficients Principle. Subsequently, another scheme must be considered.

3.3.2.2 Upwind Scheme

The upwind scheme overcomes the problems associated with the central-difference scheme by assuming that the value of the field variable, ϕ, is equal to the value of ϕ at the grid point upwind from that element. Stated differently, $\phi_e=\phi_p$ if $F_e>0$ and $\phi_e=\phi_e$ if $F_E<0$. This assumption represents the flux on a face (say, the "E" side) as $F_E = \phi_P\langle\langle F_E,0\rangle\rangle - \phi_e\langle\langle -F_E,0\rangle\rangle$, where $\langle\langle x,y\rangle\rangle$ indicates the greater of x and y.

Equation (3-19) is now expressed as

$$a_P\phi_P = a_E\phi_e + a_W\phi_W \tag{3-27}$$

with: $a_E = D_E + \langle\langle -F_E,0\rangle\rangle$

$a_W = D_w + \langle\langle F_w,0\rangle\rangle$

$a_p = a_E + a_W + (F_E - F_W)$

Equation (3-27) ensures that the coefficients cannot be negative, satisfying the constant coefficients criteria. There are, however, other limitations of the upwind scheme. In order to explain these clearly, an analytical solution to Equation (3-18) must first be considered.

3.3.2.3 Exact Solution

The 1D convection-diffusion equation,

$$\frac{d}{dx}(\rho u\phi) = \frac{d}{dx}\left(\Gamma\frac{d\phi}{dx}\right)$$

can be solved analytically. If $\phi(x=0)=\phi_0$ and $\phi(x=L)=\phi_L$, then its solution is

$$\frac{\phi-\phi_o}{\phi_L-\phi_o} = \frac{\exp\left(\frac{x\,Pe}{L}\right)-1}{\exp(Pe)-1}, \tag{3-28}$$

where Pe is the Péclet number defined as $Pe = \frac{\rho u L}{\Gamma}$. The implications of this development are illustrated in Figure 3-1, showing the spatial dependence of ϕ for various Péclet numbers. Figure 3-1 shows a linear profile as Pe$\rightarrow$0, indicating the absence of convection (i.e., pure diffusion). For Pe > 0 (corresponding to flow in the positive direction), ϕ is shifted toward ϕ_o. This indicates that the flow is becoming increasingly influenced by the flow that is upstream from ϕ. For Pe >> 0, the flow changes little from ϕ_o (the assumption of the central-difference scheme). The picture is reversed for negative Péclet numbers.

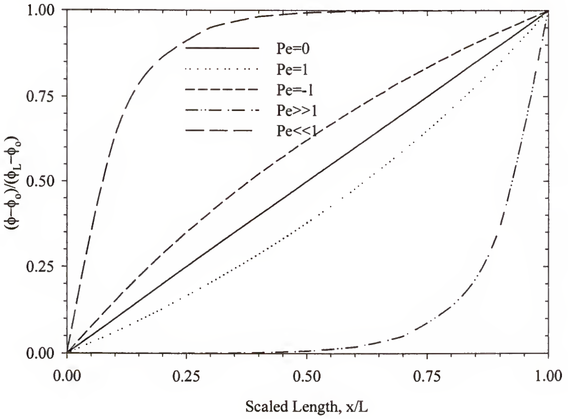


Figure 3-1 Exact solutions to Equation (3-28) for various Péclet numbers. This solution demonstrates the limitation of the upwind scheme of representing convection because a linear variable profile is only reached when Pe$\approx$0.

This demonstrates the limitations of the upwind scheme. The upwind scheme, as it is presently constructed, requires a linear profile of ϕ across the control volume. As seen in

Figure 3-1, this is only true for $|Pe| \approx 0$. As Pe becomes increasingly large, the profile of ϕ becomes more non-linear and the upwind scheme overestimates $|Pe|$.

3.3.2.4 Hybrid Scheme

Figure 3-1 also shows that there are certain Pe regimes at which the convection-diffusion equation obeys a central-difference scheme and another where the upwind scheme is more accurate. Because of this, a combination of these two schemes, called the "hybrid scheme" is used in this work.

To proceed, the conservation law, Equation (3-18), is now recast in terms of "total momentum flux". The flux, J, is defined as

$$J = \rho u \phi - \Gamma \frac{d\phi}{dx}. \qquad (3\text{-}29)$$

This transforms Equation (3-18) into

$$\frac{dJ}{dx} = 0. \qquad (3\text{-}30)$$

If a computational domain is discretized according to Figure 3-2,

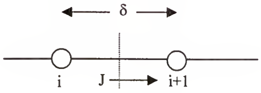


Figure 3-2. Sample discretization of 1D problem

then a modified flux, J*, can be defined through manipulation of Equation (3-29) giving

$$J^* \equiv \frac{J\delta}{\Gamma} = Pe - \frac{d\phi}{d\left(\frac{x}{\delta}\right)}. \qquad (3\text{-}31)$$

The field variable, ϕ, is simply a weighted sum of the values of its nearest neighbors, *viz.*

$$J^* = A\phi_i - B\phi_{i+1}, \tag{3-32}$$

where A and B are arbitrary coefficients. These coefficients can help put the discretized equations into a useful form.

3.3.2.5 Formulation of the Discretized Equations of Change

There are some interesting properties of A and B that can help with this formulation. If the diffusion is zero, then $\phi_i = \phi_{i+1}$ and J = Pe ϕ_i = Pe ϕ_{i+1}. Substituting into (3-32) gives simply

$$B = A + Pe. \tag{3-33}$$

The coefficients A and B are also asymmetric about Pe = 0. For example, if Pe is replaced by –Pe, the A and B are simply reverse the function, so that

$$A(-Pe) = B(Pe) \quad \text{and} \tag{3-34}$$
$$B(-Pe) = A(Pe) \tag{3-35}$$

In light of this fact, Equation (3-33) can be rewritten as

$$A(Pe) = B(Pe) - Pe, \tag{3-36}$$

and from Equation (3-34),

$$A(Pe) = A(-Pe) - Pe. \tag{3-37}$$

Equations (3-36) and (3-37) imply

$$A(Pe) = A(|Pe|) - Pe, \tag{3-38}$$

or, alternatively,

$$A(Pe) = A(|Pe|) + \langle\langle -Pe, 0\rangle\rangle. \tag{3-39}$$

A similar equation can be written for B by combining (3-39) with (3-33) yielding

$$B(Pe) = A(|Pe|) + \langle\langle Pe, 0 \rangle\rangle . \tag{3-40}$$

Substituting (3-38) and (3-39) into the flux Equation (3-32), the final discretized equation is obtained:

$$a_P \phi_P = a_E \phi_e + a_W \phi_w , \tag{3-41}$$

where: $a_E = D_E A(|Pe|) + \langle\langle -F_E, 0 \rangle\rangle$

$a_W = D_W A(|Pe|) + \langle\langle F_w, 0 \rangle\rangle$

$a_P = a_E + a_W + (F_E - F_w)$.

The different techniques used to incorporate convection in flow calculations are simply different functional relations of A. The central-difference scheme is with A = 1 - 0.5|Pe|; in the upwind scheme, A =1. The scheme used in this work is called the hybrid scheme and is a combination of the central-difference and upwind schemes. The hybrid scheme uses the upwind formulation from –2 < Pe < +2 and the central-difference scheme elsewhere. For the hybrid scheme, it can be shown, after much algebra, that the A function is represented as

$$A = \langle\langle 0, 1 - 0.5|Pe| \rangle\rangle . \tag{3-42}$$

3.3.2.6 Final Discretized Equations

All the tools to write the complete discretized conservation equations are now available which can account for both diffusion and convection. Although the presentation so far has been limited to a 1D or 2D formulation, the 3D equations are now presented. The development is the same for both cases. In these equations, the subscripts “t” and “b” represent “top” and “bottom.” The equation is

$$a_P \phi_P = a_E \phi_e + a_W \phi_w + a_N \phi_n + a_S \phi_s + a_T \phi_t + a_B \phi_b + b , \tag{3-43}$$

where: $a_E = D_e A(|Pe_e|) + \langle\langle -F_e, 0\rangle\rangle$

$$a_W = D_w A(|Pe_w|) + \langle\langle F_w, 0\rangle\rangle$$

$$a_N = D_n A(|Pe_n|) + \langle\langle -F_n, 0\rangle\rangle$$

$$a_S = D_s A(|Pe_s|) + \langle\langle F_s, 0\rangle\rangle$$

$$a_T = D_t A(|Pe_t|) + \langle\langle -F_t, 0\rangle\rangle$$

$$a_B = D_b A(|Pe_b|) + \langle\langle F_b, 0\rangle\rangle$$

$$a_P^0 = \frac{\rho_P^0 \Delta x \Delta y \Delta z}{\Delta t}$$

$$b = a_P^0 \phi_P^0$$

$$a_P = a_E + a_W + a_N + a_S + a_T + a_B + a_P^0$$

$$F_e = (\rho u)_e \Delta y \Delta z$$

$$F_s = (\rho u)_w \Delta y \Delta z$$

$$F_N = (\rho v)_n \Delta z \Delta x$$

$$F_S = (\rho v)_s \Delta z \Delta x$$

$$F_T = (\rho w)_t \Delta x \Delta y$$

$$F_B = (\rho w)_b \Delta x \Delta y$$

$$D_e = \frac{\Gamma_e \Delta y \Delta z}{(\delta x)_e}$$

$$D_w = \frac{\Gamma_w \Delta y \Delta z}{(\delta x)_w}$$

$$D_n = \frac{\Gamma_n \Delta z \Delta x}{(\delta y)_n}$$

$$D_t = \frac{\Gamma_t \Delta x \Delta y}{(\delta z)_t}$$

$$D_s = \frac{\Gamma_s \Delta z \Delta x}{(\delta y)_s}$$

$$D_b = \frac{\Gamma_t \Delta x \Delta y}{(\delta z)_t}.$$

The Péclet numbers are simply the flux, F, divided by the diffusivity, D: $Pe_i = \frac{F_i}{D_i}$. In the previous equations, the "0" superscript refers to the previous iterated value of the field variable.

3.4 Pressure Correction

Equation (3-43) allows the calculation of a field variable, ϕ, for a given flow field. Many times, one does not know what the flow field will be *prima facie*. It is then necessary to calculate the field variable for an unknown flow field. The strategy for accomplishing this is to account for the pressure terms in Equation (3-6). To understand how this works, one must realize that the pressure field is indirectly specified in the continuity equation. To see this, recall the continuity and Navier-Stokes equations as:

$$\nabla \bullet \vec{v} = 0 \tag{3-44}$$

$$\frac{\partial(\rho\vec{v})}{\partial t} + \rho\vec{v} \bullet \nabla(\rho\vec{v}) = -\nabla p + \mu\nabla^2\vec{v} + \rho\vec{g} \tag{3-45}$$

The development up to this point has allowed all terms to be calculated except the pressure term. Therefore, if the "correct" pressure field is accounted for in the Navier-Stokes equation, the continuity equation would be automatically satisfied. This is the goal of the pressure field calculation – to correct the velocity with the pressure field in a way that satisfies the continuity equation.

The discretization method employed in this work renders the governing equations into sums of forces acting on a control volume by its nearest neighbors. If this is done neglecting the influence of the pressure field, Equation (3-43) results. We now account for the pressure field through the addition of an extra term, the pressure correction. To calculate this pressure correction, assume the "correct" pressure, p, is a sum of a guessed pressure, p^*, and a pressure correction, p', such that $p = p^* + p'$. The pressure terms correspond to velocity terms – i.e., let $u = u^* + u'$ and $v = v^* + v'$. Since the pressure and velocity are not evaluated at the same grids points, the pressure is related to the velocity through its value at its nearest neighbors (and not at the point "p"). The pressure corrects the velocity through

$$a_E u_e = \sum_{nn} a_{nn} u_{nn} + b + (p_P - p_E) A_e , \qquad (3\text{-}46)$$

where nn indicates the nearest neighbors and A_e is the area of the eastern face of the control volume. A similar expression can be written for the guessed velocity and pressure field as

$$a_e u_e^* = \sum_{nn} a_{nn} u_{nn}^* + b + (p_P^* - p_E^*) A_e \qquad (3\text{-}47)$$

If equations (3-46) and (3-47) are subtracted (noting that $p = p^* + p'$ and $u = u^* + u'$) we obtain

$$a_e u_e' = \sum_{nn} a_{nn} u_{nn}' + b + (p_P' - p_E') A_e . \qquad (3\text{-}48)$$

We now arrive at the most important assumption of the SIMPLE algorithm; neglecting the $\sum_{nn} a_{nn} u_{nn}'$ term from this expression. This amounts to not accounting for the nearest neighbors to the pressure contribution. Had this term been included, corrected velocities, u_{nn}', would influence the pressure field. Unfortunately, these corrected velocities are themselves dependant on the pressure field. Subsequently, the inclusion of this term would require the calculation of the entire corrected velocity field at all grid points which, in turn, depends on the pressure field. This would increase the computational time significantly and reduce the propensity of the field variables to converge, as the velocity and pressure are interdependent in a more complex fashion. Although the nearest neighbor contributions are neglected from the SIMPLE algorithm, the velocity does influence the pressure through the mass source term, b. This indirect velocity effect on the pressure field is the source of the name "semi-implicit."

3.5 Practical Use of the SIMPLE Algorithm

All the tools needed to implement the algorithm are now available. The algorithms used in this work consist of the following steps:

1. The pressure field and velocity components are guessed.
2. Solve the momentum equations for new values of u, v, and w.
3. Solve the pressure correction equation for p'. Calculate p from $p = p^* + p'$.
4. Correct the velocity using Equation (3-48).
5. Solve for the temperature profile using the converged velocity field from step 4.
6. Make the corrected pressure, p, the new guessed pressure, p^* and repeat from step 2.

A flow chart of the algorithm is shown in Figure 3-3. The source code is given in Appendix C.

Comparison to experiments requires simulations to be performed over cylindrical geometries. To do this, the "blocking-off" technique was used. In this procedure, a

Cartesian geometry is used to approximate a cylinder by superimposing a cylindrical pattern on a rectangular grid. The velocities of the points outside the cylinder are forced to be zero and the temperature flux is taken to be infinite.

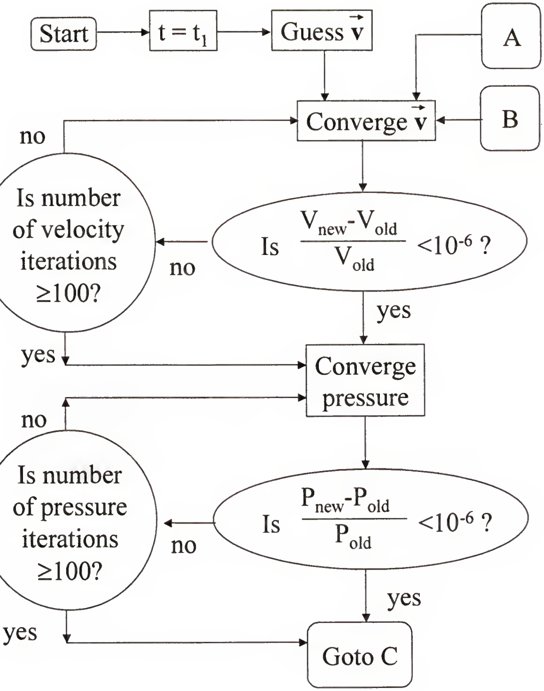

Start
$t = t_1$
Guess $\vec{v}$
A
B
Converge $\vec{v}$
Is $\frac{V_{new}-V_{old}}{V_{old}} < 10^{-6}$?
no
Is number of velocity iterations ≥100?
no
yes
yes
Converge pressure
Is $\frac{P_{new}-P_{old}}{P_{old}} < 10^{-6}$?
no
Is number of pressure iterations ≥100?
no
yes
yes
Goto C

Figure 3-3. Flow chart of finite-difference fluid flow algorithm used in this work.

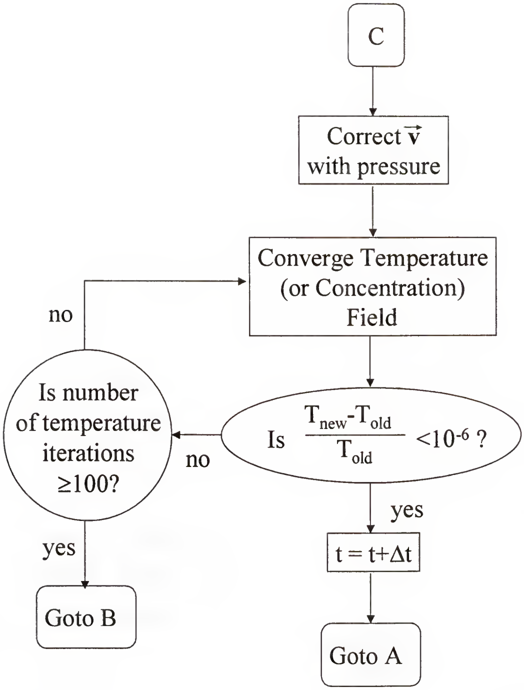

C
Correct $\vec{v}$ with pressure
Converge Temperature (or Concentration) Field
Is $\frac{T_{new}-T_{old}}{T_{old}} < 10^{-6}$?
no
Is number of temperature iterations ≥100?
no
yes
Goto B
yes
$t = t + \Delta t$
Goto A

Figure 3-3–continued.

CHAPTER 4
OXYGEN DIFFUSIVITY IN TIN AND TIN-LEAD ALLOYS

The temperature sensitivity of the diffusivity of oxygen in liquid tin and tin-lead alloys is determined using electrochemical titration in oxygen concentration cells. Such a study is important for two principle reasons. First, the oxygen tracer species used in the electrochemical flow visualization system is subject to diffusive effects over time periods greater than H^2/D (approximately 2 to 3 h). Thus, an understanding of the oxygen diffusivity is necessary. Second, oxygen diffusion studies are important undertakings in and of themselves. In many metal solidification processes, for example, oxygen incorporation in the product is a significant issue, and accordingly, the understanding of transport mechanisms (such as diffusion) is important.

The use of electrochemical cells to measure the oxygen diffusivity in liquid metals and alloys has been used for several decades since the technique was first advanced by Alcock and Bedford [Alc64] and Matsushita and Goto [Mat64]. The technique has been successfully used to study the oxygen diffusion in several liquid metals, such as indium [Hes82a, Ots84], gallium [Alc77], nickel [Ots77], iron [Ots77], copper [Obe73, Ots76, Hes82b], and silver [Ram72, Hes82b]. Specifically for tin, there have been three major studies of the oxygen diffusivity that used electrochemical measurements. Ramanarayanan and Rapp [Ram72], for example, measured the oxygen diffusivity using a radial depletion arrangement. Later, Otsuka *et al.* [Ots84] and Sears *et al.* [Sea93] conducted transient depletion experiments using a liquid bridge capped by two YSZ disks. Figure 4-1 shows the reported temperature dependency of the oxygen diffusivity data along with the experimental scatter of the results for these three studies. For the Ramanarayanan and Rapp [Ram72] and Otsuka *et al.* [Ots84] studies, there is an order of magnitude scatter in the predicted oxygen diffusivity. It is generally accepted that such inconsistencies arise from the presence of buoyancy-induced convection during measurements. As will be concluded later in this chapter, diffusive velocities are of the order of 10^{-3} to 10^{-4} cm/s, while convective velocities are typically in the $10^{1} - 10^{0}$ cm/s range (see Chapter 5); therefore, the presence of convection can lead to measured diffusivities that are significantly higher than would have been otherwise determined.

The Sears *et al.* [Sea93] study shows considerably less scatter than [Ram72] and [Ots84], due in large part to experimental refinements which reduced the possibility of thermal gradients inducing convection during the diffusivity measurements. These refinements included an axial depletion arrangement and the use of copper tubes to enclose fluid ampoules. Axial depletion experiments are less affected by convection because they can be operated in stable (*i.e.*, bottom-heavy) configurations. Radial depletion experiments, on the other hand, are intrinsically unstable [Tri00], reducing the reliability of such experiments because the possibility of convection effects. It is because of the reduced scatter in the results of Sears *et al.* [Sea93] that this method is used in this work to characterize the diffusivity of oxygen in tin and tin-lead alloys. Specifically, the goals of this part of this work are:

- To measure the oxygen diffusivity in liquid tin over the complete temperature range of the previous studies [Ram72, Ots84, Sea93] and to investigate the temperature functionality of the results.
- To analyze the results for evidence of convection during diffusion measurements.
- To measure the oxygen diffusivity in tin-lead alloys.

4.1 Ampoule Construction and Experimental Procedures

As shown in the schematic and photograph in Figure 4-2, the liquid metal is enclosed in a cylindrical tube of transparent silica (Analytical Research Systems, Gainesville, FL), with an inner diameter of 7 mm, an outer diameter of 9 mm, and of heights ranging from 3 to 8 mm. The top and bottom of the silica tube are covered by disks of 8% Yttria-Stabilized Zirconia (Cherry-O International, Amagasaki City, Japan), with a diameter of 9.0 mm and thickness of 1.3 mm. The top YSZ disk has a 0.6 mm hole in its center, which allows a wire to provide electrical contact with the working electrode. For tin, this wire was 99.97% rhenium, and for lead, 99.97% iridium was used (both 0.25 mm in diameter and from Alfa Aesar, Ward Hill, MA). The rhenium or iridium wire is partially covered by a recrystalized alumina tube with OD=3 mm, ID=2.5 mm, length=5 cm. This tube: 1) electrically isolates the lead wire from other connections; and 2) contains any metal overflow resulting from thermal expansion. The metal or alloy working electrode is either 99.999% tin (Alfa Aesar, Ward Hill, MA) or alloys of this with 99.99% lead (Allied Chemical, Morristown, NJ). All sealing is with Ceramabond 571L and 571P (Aremco Products, Valley Cottage, NY), which must be cured at 200 °C for 2 h.

Electrical contacts are made to the YSZ by attaching copper blocks (TMR Engineering, Micanopy, FL) to the top and bottom sensors (The top port is a cylinder with a radius of 1.25 cm, and a height of 2.4 cm. The bottom is a cylinder with radius 1.35 cm and height of 1 cm). These copper ports have hollowed-out regions, which contains the reference electrodes – a 50 atom % mixture of powdered Cu/Cu_2O (Alfa Aesar, Ward Hill, MA). These copper ports were the major refinement to the diffusivity measuring system advanced by [Sea93] discussed above. Because copper has a high thermal conductivity, any temperature variations are reduced and the impact on the diffusion measurement will be minimized.

The ampoule is suspended in a large recrystalized alumina tube (Vesuvius McDanel, Beaver Falls, PA) with OD of 5.72 cm, ID of 5.08 cm, and 50.0 cm in length, which allows for the evacuation of the air from the cell and the replacement with an inert Ar blanket (not shown in Figure 4-2). All temperature measurements are made with a Type R (13% rhodium-platinum vs. platinum) thermocouple calibrated with reference tables available from many reference books (such as [Ome96]).

Before construction, it is important to remove any native oxide contamination. Tin is etched for 60 s in a solution of 95% HBr and 5% Br_2 (by volume) and rinsed three times with methanol. Lead is etched with 5% HNO_3 and rinsed three times with methanol. The quartz fluid enclosures and all alumina parts are etched with a solution of 57% water, 35% HNO_3 and 8% HF for 10 minutes, rinsed with deionized water, and rinsed 3 times with methanol. The copper electrical ports are sanded to remove oxide contamination; then washed with soap and water, rinsed with thrchloroethylene, etched for 10 min in 5% HNO_3, and then rinsed with water and methanol.

When an experiment is conducted, the ampoule is placed in a furnace, where it is alternately evacuated to a vacuum of at least 100 kPa and filled back up with 99.999% argon several times to remove the residual air. For this study, the argon was further purified by passing through 2 titanium sponge packings held at 1000 °C. After purging, argon is then allowed to flow freely through the system for two hours and the temperature is slowly increased at a rate of approximately 40 °C per 10 min until 200 °C is reached.

The system is then heated at 200 °C for 2 h to cure the remaining adhesive. The temperature is then slowly increased to the experimental temperature. As the proper temperature is reached, the potential between both the bottom and top working electrode and the middle rhenium wire is expected to be approximately equal to the value of the saturation potential, which corresponds to the maximum concentration of oxygen in the fluid before solid oxides precipitate. For liquid tin, the saturation potential was determined experimentally by Ramanarayanan and Rapp [Ram72] as $E_{sat} = 746.915 - 0.28059T$, where E_{sat} is in mV, and T is in K. For reference, the saturation potential at 650 °C (a temperature typically studied in this work) is 488 mV. For tin-lead alloys, the saturation concentrations and potentials were determined by Das and Ghosh [Das72].

4.2 Experimental Interpretation

Two different types of diffusivity experiments were performed: transient and steady state. The operation and usefulness of each of these types of experiments are now explained in turn.

4.2.1 Transient EMF Measurements

Transient EMF experiments are operated in either a top or bottom depletion mode, corresponding to the side of the melt at which the oxygen in depleted. For all depletion experiments, an initial concentration equal to 50 mV above the saturation potential (usually around 0.567 V, corresponding to an oxygen partial pressure of 10^{-6} atm) is applied to both the top and bottom copper contacts and the system is allowed to equilibrate until the same potential is measured at both ends of the melt. This typically took at least 1½ to 3 hr. At the beginning of the experiment, a large voltage (typically

1.2 V, corresponding to an oxygen partial pressure of about 10^{-12} atm) is applied to either the top or bottom cell, thereby depleting oxygen at that electrolyte-melt interface. As the oxygen is depleted, oxygen from the remainder of the tin transports toward the depleting electrode, in a process limited by the diffusion of oxygen. As the experiment proceeds, the open circuit EMF at the opposite side of the working electrode is measured with time and analyzed with Equation (2-24).

4.2.2 Steady State Current Measurements

Steady-state current measurement experiments are performed by applying either a stabilizing (bottom-heavy) or destabilizing (top-heavy) concentration gradient across the melt and measuring the steady current produced on each side of the melt, which would typically range from 1 μA to 1 mA. The experiment is taken to be "steady state" when the measured currents at both sensors are the same to within 10 %. Current is then related to the diffusivity through Equation (2-25). As detailed in Chapter 6, this procedure can be useful in quantifying the effect of convection on diffusivity measurements, as the concentration gradient applied across the melt can be related to a solutal Rayleigh number.

4.3 Results and Discussion

Over 1,000 individual transient experiments were performed for different fluid aspect ratios (i.e., heights) and for top and bottom depletion operation. The data set was reduced by eliminating experiments whose theoretical and experimental intercepts deviated by more than ±30%. The selected results of the transient measurements are shown in Figure 4-3 along with a stabilizing and destabilizing steady state experiment. If the results are fit to an Arrhenius model, the following is obtained:

$$D\left(\frac{\text{cm}^2}{\text{s}}\right) = 5.98(\pm 0.88)\times 10^{-3}\exp\left(\frac{-3900\pm 750}{T}\right) \tag{4-1}$$

Upon examining Figure 4-3, some initial conclusions can be made. The top depletion experiments (filled symbols) tend to have larger diffusivity values than bottom depletion experiments (unfilled symbols). In fact, if the top and bottom depletion experiments are regressed separately, the following expressions are obtained:

$$D_{\text{top}}\left(\frac{\text{cm}^2}{\text{s}}\right) = 6.63(\pm 0.87)\times 10^{-3}\exp\left(\frac{-3900\pm 700}{T}\right) \tag{4-2}$$

$$D_{\text{bottom}}\left(\frac{\text{cm}^2}{\text{s}}\right) = 4.96(\pm 0.90)\times 10^{-3}\exp\left(\frac{-3800\pm 800}{T}\right) \tag{4-3}$$

From these equations, the top depletion experiments predict diffusivities that average 50% larger than the bottom depletion experiments. This is significant because convection, if present, would be manifested in top depletion arrangements, and as discussed previously, convection should increase the measured diffusivity. It would appear, subsequently, that the experiments conducted show convective effects. Figure 4-3 also shows the results from a stabilizing and destabilizing steady state experiment. Again, the result for the potentially unstable configuration is about 54% larger than for the stable arrangement (1.5×10^{-4} cm^2/s for bottom heavy vs. 1.7×10^{-4} cm^2/s for top heavy). If the results from this study are compared with previous tin diffusion studies [Ram72, Ots84, Sea93] as in Figure 4-4, it can be seen that the scatter of the measurements in this study is comparable to that of Sears *et al.* [Sea93] and is much better than either the Ramanarayanan and Rapp [Ram72] or Otsuka *et al.* [Ots84] studies. Moreover, this study has included measurement over the entire temperature range of all three of these studies. Thus, although the results of this study seem to be significant contributions to the oxygen diffusion problem, convective effects may be present nonetheless.

To analyze convective effects further, the effect of the fluid aspect ratio was analyzed for top and bottom depletion experiments. The results in Figure 4-5 show that, for top depletion experiments, there is an ordered dependence of the diffusivity with height over the entire temperature range. This order is precisely what is expected for experiments performed in the presence of a convecting fluid since $Ra \sim h^3$. Noting that this height functionality does not exist for bottom depletion experiments reinforces this conclusion.

This study has shown that despite efforts to reduce thermally induced convection, their effects are nonetheless present. There are two other possible causes of convection: (1) solutal-induced buoyancy; and (2) tilting of the ampoule. Explanation (1) is considered in Chapter 6. Explanation (2) is now expanded upon in more detail.

If a diffusivity ampoule were tilted with respect to gravity, instabilities will propagate through the fluid at lower Rayleigh numbers (see Section 5.7). Therefore, convection may be present at Rayleigh numbers significantly lower than the first critical Rayleigh number for the untitled cases. To demonstrate this, a calculation was performed using the algorithm developed according to Chapter 3 for a cubical container tilted 10° in one azimuth. The boundary conditions were those which would be encountered for a typical diffusivity experiment: no-slip and no-flux conditions at all surfaces. The result for Ra=2,000 is shown in Figure 4-6 showing cellular convection manifested by two corner cells. A similar calculation for 0° shows no convection.

The effect on diffusion experiments can be seen by examining streamline traces in Figure 4-6 whose convective velocities are about 0.5 cm/s; therefore, if some oxygen is located (for example) at the bottom of the container, the convection will transport oxygen very quickly from the bottom to the top of the first cell. The oxygen then diffuses across streamlines. At this point, the oxygen packet is quickly transported to the top of the cell and the top of the container. In effect, the presence of convection has effectively reduced the height of the fluid from the container height to the distance between the convection cells. The quantify the effect on the measured diffusivity, consider the time it would take for a packet of oxygen to transport from the bottom to the top in the tilted fluid in Figure 4-6, if the effective diffusive height is assumed to be 10% of the non-tilted case.

Before leaving the topic of tilt, two comments should be made. First, a tilt of 10° is not exceptionally excessive. Care was taken when constructing ampoules to avoid tilting; however, a 10° tilting was difficult to avoid. Thus, if more accurate diffusion studies are needed, either: (a) strict control of the tilting of the fluid ampoule must be followed; or (b) microgravity investigations should begin. Second, the calculation in Figure 4-6 was performed for tilting in one direction only. If a tilting in the second azimuth was to be considered, an even more complicated response would be expected.

4.4 Alternate Temperature Models

The temperature effect of the diffusivity is also an outstanding issue resulting from the lack of understanding of the diffusion process in liquid metals. It is widely accepted that diffusion in solids is an activated process requiring a critical energy to initiate diffusion (and resulting in an Arrhenius temperature dependency of the diffusivity); furthermore, molecular modeling has made it equally accepted that diffusion in a gas is not an Arrhenius process, instead governed by a power law. The kinetic theory model of Hirschfelder [Hir54], for example, represents the diffusivity as a function of $T^{1/2}$. Diffusion in a liquid, whether self-diffusion or the diffusion of a tracer or contaminate, is not as well-understood as that in a gas or solid. In practically all previous studies of the oxygen diffusivity in liquid metal systems, authors have simply assumed an Arrhenius temperature dependence and all results obtained were regressed to this model as was done in the previous sections. There is, however, alternate models of liquid diffusion based on rather simple geometrical arguments that will now be explored. The model of Swalin [Swa59], for example, is based on random density fluctuations in the liquid metal which create voids between adjacent atoms through which the atom may diffuse if the void is large enough. Such a model predicts diffusivities that are related to the temperature as T^2. A similar model was proposed by Cohen and Turnbull [Coh59], assuming atoms to be solid spheres moving in a cage to prevent interatomic contact. Diffusion is assumed to be the movement of this "caged" atom through an interatomic void; diffusion is mechanistically explained as the redistribution of the interatomic voids as the atom diffuses. Such a treatment results in a diffusivity that is related to $T^{\frac{1}{2}}$. Finally, to complicate things further, a review by Nachtrieb [Nac67] of self-diffusion data for liquid metals suggests that for some metals the T^2 model may be more accurate, but others may be governed by the Arrhenius model, and none to be governed by the $T^{\frac{1}{2}}$ model. Nachtrieb also even goes so far as to suggest that there may not be a simple temperature functionality of the diffusivity. To help settle this disagreement, this study will analyze the oxygen-tin diffusivity results and test the data against these two models and the Arrhenius model. Additionally, since both the Swalin and Cohen and Turnbull models are simply power law representations of the temperature functionality, a generalized power law model is also analyzed which represents the diffusivity as D=AT^n. It should be noted at this point, however, that this generalized power law is intended only to generalize [Swa59] and [Coh59] – no physical justification of this model is implied.

When the pure tin results are regressed to these alternate temperature models, the results are given in Table 4-1.

Table 4-1. Diffusivity regressions for pure tin data using alternate temperature models.

Model[1]	A	B	n
Swalin [Swa59]	$2.7\ (\pm 2.1) \times 10^{-11}$	$1.0\ (\pm 0.87) \times 10^{-4}$	----
Cohen & Turnbull [Coh59]	$2.5\ (\pm 1.9) \times 10^{-11}$	$6.4(\pm 5.6) \times 10^{-4}$	----
General Power Law	$1.5(\pm 1.3) \times 10^{-21}$	----	0.28

[1] For Swalin, $D = AT^2 + B$

For Cohen & Turnbull, $D = AT^{1/2} + B$

For General Power Law, $D = AT^n$

The predicted oxygen diffusivities for each of these models, combined with the Arrhenius model are shown in Figure 4-7 and demonstrate that all four models predict diffusivities equally well within the experimental scatter encountered in this work. This is especially significant when comparing the general power law model with the others as this model has no physical basis and yet represents the data as well as the other three which do possess physical justification. This result suggests that either: (1) the temperature range needs to be further expanded to determine the correct thermal functionality of the diffusivity; or (2) even more accurate diffusivity measurements are needed. As will be discussed in Chapter 7, however, the temperature range studied in this work represents the limit of usefulness of electrochemical techniques for diffusivity measurements. Therefore, future researchers interested in the temperature functionality of the diffusivity must have the most accurate measurements conceivable, further supporting the need for microgravity studies.

4.5 Tin-Lead Alloys

Electrochemical measurements are especially useful for oxygen diffusivity studies in alloys because the dependence of the diffusivity with alloy concentration exhibits a wide range of behaviors. For example, the study of El-Naggar and Parlee [Eln71] for copper alloys shows diffusivity-concentration functionalities that either change quickly (e.g., for Cu-Pt) or exhibit no discernible functionality (e.g., for Cu-Ni). Furthermore, the study of Hahn and Stevenson [Hah77] for the gallium-indium system showed alloy diffusivities that did not change significantly with intermediate alloy concentrations, yet differed from either of the pure component diffusivity by an order of magnitude, suggesting significant effects on the diffusivity at low alloy concentration. Such behaviors result from the complex effect of alloy mixing on transport (and thermodynamic) properties that are not well understood. In this study, the oxygen diffusivity in tin-lead alloys is measured with a galvanometric measurement technique. Emphasis is on the temperature and alloy concentration effect. To analyze the concentration functionality, an empirical alloy diffusivity model of Perkins and Geankoplis [Per69] is used.

The diffusivity of pure tin was detailed previously and compares well with the results of Ramanarayanan and Rapp [Ram72], Otkuka *et al.* [Ots84], and Sears *et al.* [Sea93] in Figure 4-3 and showed good agreement. When experiments for pure lead are performed, the following Arrhenius temperature dependence is obtained:

$$D_{Pb}\left(\frac{cm^2}{s}\right) = 3.58(\pm 0.70)\times 10^{-3}\exp\left(\frac{-3300 \pm 125}{T}\right) \quad (4\text{-}4)$$

When the lead results are compared with Homna *et al.* [Hom71] (who made axial current measurements), Szwarc *et al.* [Szw72] (radial current measurements), and Otsuka and Kozuka [Ots75] (axial EMF measurements) in Figure 4-8, the agreement is not as good as was observed for tin, the results of this study predicting the highest diffusivities. The results of the lead studies differed from one-half to a full order of magnitude. To the extent that the results of this study for pure tin compare acceptably to previous researchers leads to the conclusion that there is no systematic cause of convection in the experimental system used in this work (assuming the measurements of previous researchers were free of convection!). The results obtained for lead simply underscore the need to study this system more closely.

When the alloy diffusivities are fit to an Arrhenius model, the following are obtained:

$$D_{10\%Sn}\left(\frac{cm^2}{s}\right) = 3.47(\pm 0.20)\times 10^{-3}\exp\left(\frac{-4000 \pm 430}{T}\right) \quad (4\text{-}4)$$

$$D_{90\%Sn}\left(\frac{cm^2}{s}\right) = 3.25(\pm 0.73)\times 10^{-3}\exp\left(\frac{-4375 \pm 100}{T}\right) \quad (4\text{-}5)$$

The activation energies resulting from the Arrhenius fits are shown in Table 4-2 as a function of alloy concentration and top or bottom depletion operation. The activation energies are remarkably constant over the range explored in this work. Indeed, the top depletion activation energies are especially invariant, the minimum and maximum value being different by only 12% and the standard deviation of the data being 2.8 kJ/(mol-K).

In the bottom depletion experiments, the X_{Sn}=0.9 case is significantly lower than the other cases, making the minimum to maximum deviation 55% and a standard deviation of 9.5 kJ/(mol-K). If this point is taken to be an outlier, the minimum/maximum separation is 26% and the standard deviation becomes 5.45 kJ/(mol-K).

Finally, the activation energies obtained in Table 4-2 are comparable with other activation energies obtained for other alloys. For example, Hahn and Stevenson [Hah77] reports activation energies for $Ga_{0.1}In_{0.9}$ and $Ga_{0.95}In_{0.05}$ to be 43.4 kJ/(mol-K) and 39.1 kJ/(mol-K) respectively.

Table 4-2. Activation Energies for oxygen diffusivity in tin-lead alloys

X_{Sn}	Activation Energy, E_{act} Top Depletion kJ / (mol-K)	Activation Energy, E_{act} Bottom Depletion kJ / (mol-K)
0	24.8	31.2
0.1	33.3	26.8
0.9	36.4	16.2
1	32.4	36.2

The effect of alloy concentration on the oxygen diffusivity is shown in Figure 4-9. The alloy diffusivity shows a decrease with alloy concentration from either the pure elemental diffusivities, an observation not observed for other alloys. For example, the In-Ga alloy investigation by Hahn and Stevenson [Hah77] shows oxygen diffusivities that, like the results of this study, are relatively constant and change at low alloy concentration. The diffusivity for In-Ga, however, lies between that of the pure elements.

For the Sn-Pb system in this work, the diffusivities of both alloy concentrations were lower than for pure tin and lead.

The differences between the pure elements and alloys observed for tin-lead are not as drastic as those previously observed. The differences between the diffusivities of the pure metal and its corresponding metal-rich alloy are about one-third of an order of magnitude for both tin and lead. To compare, Hahn and Stevenson [Hah77] observed a nearly a full order of magnitude increase in diffusivity between 100% indium and 95% indium / 5% gallium and El-Naggar and Parlee [Eln71] noted an increase of over an order of magnitude between 100% copper and 98% copper / 2% platinum.

The results in Figure 4-8 can be explained using an empirical model of multicomponent diffusion of Perkins and Geankoplis [Per69]:

$$\log\left(D_a\eta_a^{1-\varepsilon}\right) = X_1\log\left(D_1\eta_1^{1-\varepsilon}\right) + X_2\log\left(D_2\eta_2^{1-\varepsilon}\right), \tag{4-6}$$

where D is the diffusivity, η is the viscosity, "a" indicates the alloy properties, and "1" and "2" are the properties of the alloy components. This model is physically based on an argument of atoms jumping from position to position in the mixture. The parameter ε is taken as an adjustable parameter in the model, although it is physically related to Gibbs energy of mixing of the solution [Ola63]. The use of Equation (4-6), itself, requires liquid metal viscosity data (taken from Hirai [Hir93]) as well as a model for the viscosity of liquid metal alloys. The model chosen was that of Chhabra [Chh93, Chh95] and represents the alloy viscosity as

$$\log(\eta_a + 1) = 10^{b_1} T^{b_2}, \tag{4-7}$$

where b_1 and b_2 are given in [Chh93]. The viscosity model in Equation (4-7) is based on mixing thermodynamic arguments and the b_1 and b_2 parameters are regressed from experimental data. For Sn-Pn, the b_1 and b_2 parameters are given by [Chh95]. In the model in Equation (4-6), an optimized value of the parameter ε=2.2 was used. Perkins and Geankoplis [Per69] found ε=0.2 to best represent the data they were examining; yet they did not attempt to apply this model to liquid metals. The alloy results are compared with this model in Figure 4-8 and show good agreement.

4.6 Conclusion

The oxygen diffusivity in liquid tin and tin-lead alloys is measured over the largest temperature range studied to date. In the results for pure tin, evidence of convection was detected by comparing the measured diffusivities for top and bottom depletion experiments and by examining the dependence of the diffusivity to fluid height. Additional power law models of the thermal sensitivity of the diffusivity were investigated and demonstrated that the Arrhenius model predicts diffusivities that are no better than power law models. Diffusivities in tin-lead alloys are also measured and it is determined that the diffusivities for the pure metals are both higher than their corresponding 90% alloys. This phenomenon has not been observed for other alloy systems, but has been shown to be consistent with a multicomponent diffusion model based on thermodynamic mixing rules.

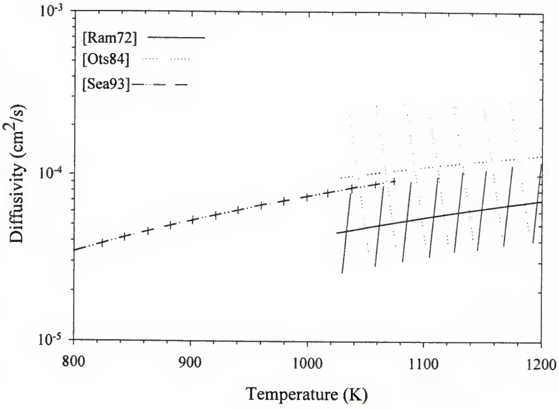
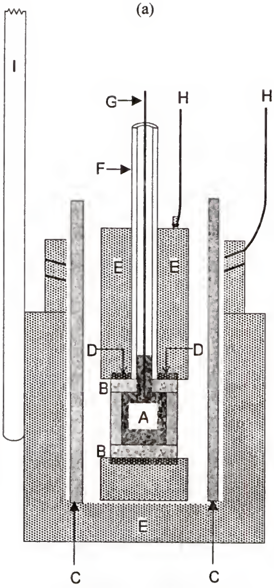


Figure 4-1. Previous diffusivity measurements for 100% tin. The vertical lines indicate the scatter reported in by the study.

(a)



(b)

Key for Figure 4-2(a)

A	Liquid Tin or Tin-Lead Alloy (working electrode)
B	YSZ Electrolyte
C	Fused silica spacer
D	50 mole % mixture of powdered Cu/Cu_2O (reference electrode)
E	Copper ports (used to make electrical contacts)
F	Alumina overflow tube
G	Melt contact wire (rhenium for Sn and 90% Sn; iridium for Pb and 90% Pb)
H	Copper lead wires
I	Type R thermocouple

Figure 4-2 Experimental diffusivity measurement system
(a) Schematic (b) Photograph of fluid ampoule

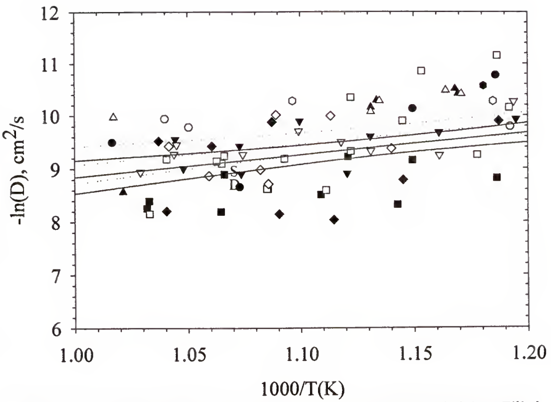


Figure 4-3. Diffusivity trends for pure tin. The symbol indicates heights. Filled symbols = top depletion experiments, Unfilled symbols = bottom depletion experiments. S, D = Steady state experiments with Stabilizing and Destabilizing concentration gradient.

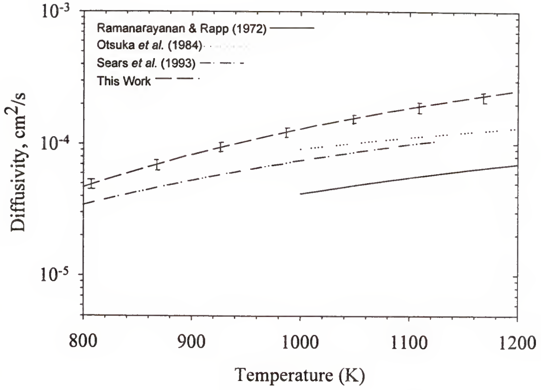
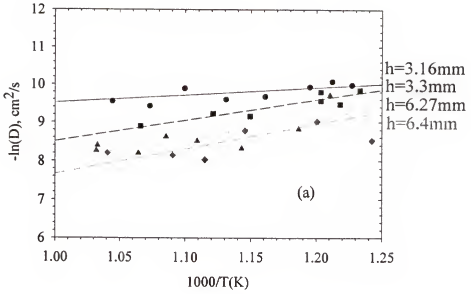


Figure 4-4. Comparison of this work to previous tin experiments. Error bars indicate experimental scatter.



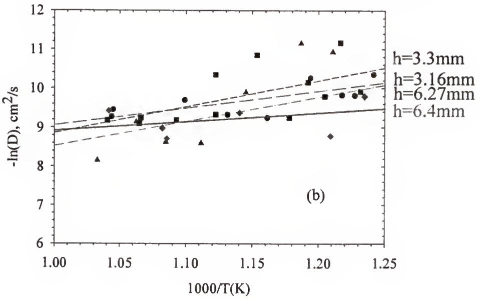


Figure 4-5. Height dependence of (a) top depletion experiments; and (b) bottom depletion experiments. The exact height relation for top and not bottom depletion experiments suggests convection.

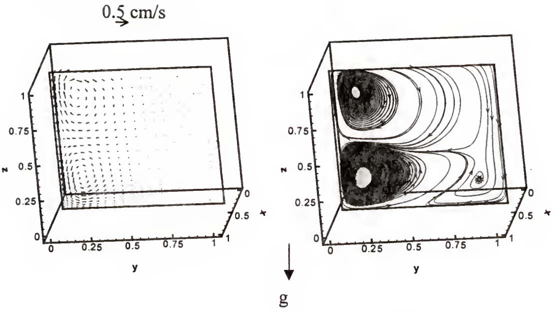


Figure 4-6 Computational fluid dynamics simulation with 10° tilt of domain. Calculations conducted for a perfect cube (γ=1); all surfaces are no-slip and adiabatic.

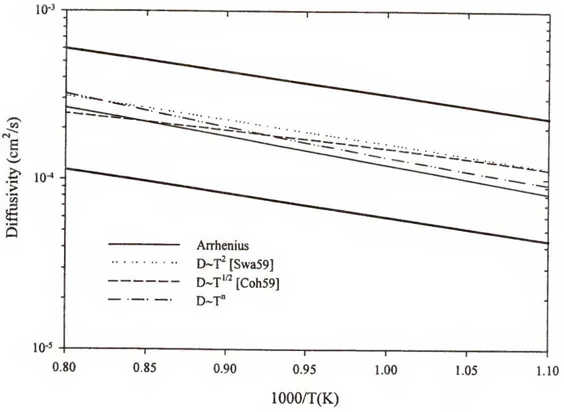


Figure 4-7 Comparisons of thermal models studied in this work. The bold line indicates the experimental scatter in the data encountered in this work.

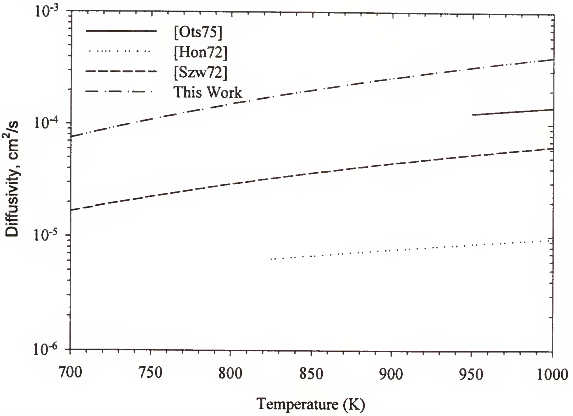


Figure 4-8. Comparisons of pure lead diffusivity with previous work.

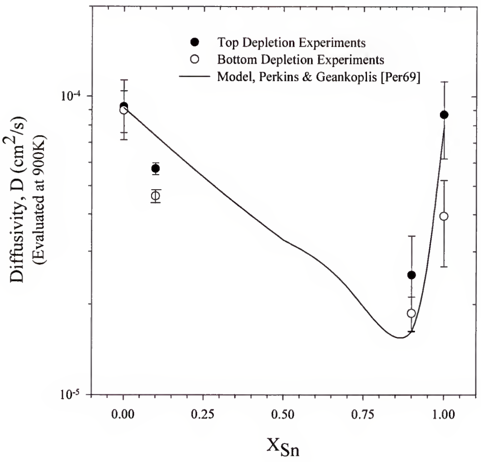


Figure 4-9 Oxygen diffusivity in tin-lead alloys at T=900K. Results shown with multicomponent diffusion model of [Per69] with ε=2.2.

CHAPTER 5
BUOYANCY-DRIVEN FLOWS IN ENCLOSURES HEATED FROM BELOW

Flow bifurcations in low Prandtl number fluids are interesting not only to calibrate the electrochemical visualization technique in this work, but is also an important fluid mechanics problem in and of itself. The first transition from conduction to steady convective flow (the first critical Rayleigh number) is independent of Prandtl-number (i.e., of fluid type), and has been studied by several authors including [Cha61, Cha70, Cha71, Cat72]. The oscillatory bifurcation, however, <u>does</u> depend on Prandtl-number, and its characterization as a function of container aspect ratio has not hitherto been systematically characterized for fluids with very low Prandtl-numbers. It is the second critical Rayleigh number transition that is the focus of this chapter. To motivate these results, a few moments are taken to explore the mathematical techniques available for studying Rayleigh-Bénard convection and the reasoning behind the selection the numerical technique selected for this work.

5.1 Linear Stability Techniques for Predicting Ra_{c1}

The linear stability technique has been useful in characterizing the first critical Rayleigh number transition. The method, formalized by Chandrasekhar [Cha61], analyzes linear disturbances of the momentum and energy conservation equations, neglecting cubic and higher ordered instabilities. Although the details of the Chandrasekhar formulation will be omitted, Chandrasekhar makes predictions of Ra_{c1} for infinite aspect ratio enclosures and various boundary conditions. These predicted values (in Table 5-1) are used as important asymptotic comparisons to the results in this work.

Other linear stability analyses have been conducted for finite aspect ratios. First critical Rayleigh number analyses of Cartesian geometries were performed by Catton and Edwards [Cat70], and a similar method was used by Rosenblat [Ros82] for cylindrical geometries. Linear stability studies provide a useful method of calculating the first critical Rayleigh number and the resulting flow patterns; however, they suffer from the distinct disadvantage that they are unable to predict flow configuration in the supercritical regimes (*i.e.*, at the second critical Rayleigh number). Oscillating flows are, by definition, non-linear events; and as such, non-linear stability techniques must be developed for the analysis of such flows. The mathematical formalism of non-linear stability is currently a very active area of research, and no single technique has yet been accepted as superior. Therefore, one must turn to numerical techniques to make these predictions, and this is the strategy adapted in this study

Table 5-1. First Critical Rayleigh numbers calculated from Linear Stability [Cha61]

Boundary Conditions	Ra_{c1}
Free-Free	693
Fixed-Free	1101
Fixed-Fixed	1907

5.2 Numerical Studies

Most numerical studies of the transition to oscillatory flow have been limited to two-dimensional enclosures. For example, Ozoe and Hara [Ozo95] characterize the transition to periodic oscillations in fluids of $Pr = 0.01$ in two-dimensional rectangular enclosures of aspect ratio (length/height) of 4.0 using a finite-difference scheme. Once convection began, the calculated oscillations manifested themselves in the form of multiple convection cells that deform as time progressed. They also conclude that the difference between the first and second critical Rayleigh numbers decreases as the Prandtl number decreases. Unfortunately, because they also conclude that the initiation of oscillations depends on grid size, the reliability of these results is debatable.

Two-dimensional studies of the transition to oscillatory flow, however, often oversimplify the rich convection which may be present, especially those in Cartesian enclosures. With Cartesian co-ordinates, the presence of the corners can contribute significantly to the overall flow pattern. Three-dimensional convection, whether oscillatory or not, often has vortices in the corners of the domains [Gup98, Cru99], and this effect cannot be reproduced by two-dimensional algorithms, therefore a three-dimensional numerical study is needed to properly determine the second critical Rayleigh number.

One of the very few three-dimensional numerical studies of natural convection of low-Prandtl-number fluids was performed by Neumann [Neu90]. This study considered the natural convection of low-Prandtl-number fluids (Pr=0.02) in vertical circular cylinders heated from below, in both steady and periodic oscillatory regimes. The results show that in the steady regime, multiple steady states are possible under similar conditions depending on the past history of the fluid. This work did resolve the fundamental process of oscillation in low Prandtl-number liquids from that observed in gases (Pr=1). Oscillations of gases in cylindrical enclosures are accompanied by the rotation of the plane of symmetry about the axis of the cylinder, while in liquids it involves spatial changes in the flow structure. Unfortunately, the time-dependent simulations of Neumann [Neu90] predict second critical Rayleigh numbers for gallium (Pr=0.02) that are much below those observed experimentally by Müller *et al.* [Mul84]. Another unexpected result was obtained for oscillatory calculations of low-Pr fluids in the numerical study of Nakano *et al.* [Nak98], conducted for a rectangular enclosure with an aspect ratio of 5.0. In this study, periodic oscillations were observed, but at later computational time steps, the fluid began to behave chaotically, leading to algorithm-dependent results. The results of this work extend those of Nakano *et al.* [Nak98] to lower aspect ratio enclosures that are more important for crystal growth experimenters.

5.3 Conservation Equations

The conservation equations are now developed that are be used with the numerical algorithm presented in Chapter 3. The first step is to assume that the modeled fluid obeys the Boussinesq approximation – that the fluid properties are constant, except for a linear variation of the density with temperature in the body force term (*e.g.*, the last term in Equation 3-6). Additionally, compressibility effects, viscous dissipation, and solutal inhomogenities are neglected. The conservation equations in vector form are then [Bir60]

$$\text{CONTINUITY} \qquad \frac{\partial \rho}{\partial t} + \nabla \bullet (\rho \vec{v}) = 0 \tag{5-1}$$

$$\text{MOMENTUM} \qquad \rho \frac{\partial \vec{v}}{\partial t} + \rho(\vec{v} \bullet \nabla \vec{v}) = \mu \nabla^2 \vec{v} - \nabla p + \rho \vec{g} \tag{5-2}$$

$$\text{ENERGY} \qquad \rho C_p \frac{\partial T}{\partial t} + \rho C_p (\vec{v} \bullet \nabla T) = k \nabla^2 T + 2\mu \vec{\vec{D}} : \vec{\vec{D}}\,. \tag{5-3}$$

Following the Boussinesq approximation [Bou03], the density is a linear function of temperature, so expressing the density as a Taylor series gives

$$\rho = \rho_o + \left.\frac{\partial\rho}{\partial T}\right|_{T_o} (T - T_o); \tag{5-4}$$

or, rearranging,

$$\frac{\rho}{\rho_o} = 1 + \frac{1}{\rho_o} \left.\frac{\partial\rho}{\partial T}\right|_{T_o} (T - T_o). \tag{5-5}$$

The quantity $\frac{-1}{\rho_o}\left(\frac{\partial\rho}{\partial T}\right)\Big|_T$ is, by definition, the thermal expansion coefficient, β.

The following dimensionless variables are now introduced:

$$\vec{v}^* = \frac{\vec{v}H}{\nu} \tag{5-6}$$

$$T^* = \frac{T - T_c}{T_h - T_c} \tag{5-7}$$

$$\nabla^* = H\nabla \tag{5-8}$$

$$t^* = \frac{t}{\hat{t}} = \frac{t}{H^2/\nu}; \tag{5-9}$$

where H is the fluid height, ν is the kinematic viscosity, T_c is the cooler temperature and T_h is the hotter temperature. Equation (5-5) is now

$$\frac{\rho}{\rho_o} = 1 - \beta\Delta T T^*, \tag{5-10}$$

where $\Delta T = T_H - T_C$. Substituting these values into the continuity equation yields

$$\frac{\partial}{\partial t}\left[\rho_o\left(1 - \beta\Delta T T^*\right)\right] + \nabla \bullet \left[\rho_o\left(1 - \beta\Delta T T^*\right)\vec{v}^* \frac{\nu}{H}\right] = 0; \tag{5-11}$$

or, alternatively,

$$\frac{\partial}{\partial t}\left(1 - \beta\Delta T \rho_o T^*\right) + \nabla \bullet \left(\rho_o \vec{v}^* \frac{\nu}{H}\right) - \nabla \bullet \left(\beta\Delta T \rho_o T^* \vec{v}^* \frac{\nu}{H}\right) = 0. \tag{5-12}$$

Scaling the time and the ∇ operator yields

$$\frac{H}{\upsilon\hat{t}}\frac{\partial}{\partial t^*}\left(-\beta\Delta TT^*\right)+\nabla^*\bullet\vec{v}^*-\nabla^*\bullet\left(\beta\Delta TT^*\vec{v}^*\right)=0. \quad (5\text{-}13)$$

For a Boussinesq fluid, $\beta\Delta T$ must necessarily be small, and the continuity equation becomes

$$\nabla^*\bullet\vec{v}^*=0. \quad (5\text{-}14)$$

This is the same form of the continuity equation as for incompressible fluids.

Substituting the scaling factors into the momentum equation yields

$$\frac{\rho_o-\rho_o\beta\Delta TT^*}{\hat{t}}\frac{\partial\vec{v}^*}{\partial t^*}+\frac{\rho_o-\rho_o\beta\Delta TT^*}{H}\frac{\nu^2}{H^2}\vec{v}^*\bullet\nabla^*\vec{v}^*=\frac{\mu}{H^2}\frac{\nu}{H}\nabla^{*2}\vec{v}^*-\frac{\hat{p}}{H}\nabla^*p^*\dots\dots$$

$$\dots\dots+\vec{g}\left(\rho_o-\rho_o\beta\Delta TT^*\right), \quad (5\text{-}15)$$

where the parameter $\hat{p}$ scales the pressure. Again, $\beta\Delta T$ is neglected everywhere except in the body force term. Doing this, and dividing the result by $\frac{\nu\mu}{H^3}$ gives

$$\frac{\partial\vec{v}^*}{\partial t^*}+\vec{v}^*\bullet\nabla^*\vec{v}^*=\nabla^{*2}\vec{v}^*-\frac{\hat{p}}{H}\nabla^*p^*+\frac{\vec{g}\rho_oH^3}{\mu\nu}-\left(\frac{\vec{g}\rho\beta\Delta TH^3}{\mu\nu}\right)T^*. \quad (5\text{-}16)$$

At this point, the $\frac{\hat{p}}{H}\nabla^*p^*$ and $\frac{\vec{g}\rho_oH^3}{\mu\nu}$ terms can be combined into a single potential gradient term, called the modified pressure, p^{**}. Notice also that the quantity in parentheses is the ratio of the Rayleigh to the Prandtl numbers. Finally, express the gravity vector as $\vec{g}=g\vec{k}$, where $\vec{k}$ is the unit vector in the direction of gravity. Combining these into the momentum balance equation gives

$$\frac{\partial\vec{v}^*}{\partial t^*}+\vec{v}^*\bullet\nabla^*\vec{v}^*=-\nabla^*p^{**}+\nabla^{*2}\vec{v}^*-\frac{Ra}{\Pr}T^*\vec{k}. \quad (5\text{-}17)$$

For the energy conservation equation, the viscous stresses are neglected, and the scaled parameters are substituted, giving

$$\rho_o\left(1-\beta\Delta T T^*\right)\frac{C_p}{\hat{t}}\Delta T\frac{\partial T^*}{\partial t^*}+\rho_o C_p\frac{\left(1-\beta\Delta T T^*\right)}{H}\frac{\nu}{H}\vec{v}^*\bullet\nabla^* T^*=\frac{k}{H^2}\Delta T\nabla^{*2}T^*.$$

(5-18)

Simplifying the algebra yields the final form as

$$\Pr\left(\frac{\partial T^*}{\partial t^*}+\vec{v}^*\bullet\nabla^* T^*\right)=\nabla^* T^*. \tag{5-19}$$

The analogous species concentration equation can now be written by inspection as

$$Sc\left(\frac{\partial c^*}{\partial t^*}+\vec{v}^*\bullet\nabla^* c^*\right)=\nabla^* c^*, \tag{5-20}$$

where Sc is the Schmidt number, defined as $\frac{\nu}{D}$, where D is the molecular diffusivity.

For this work, the Cartesian enclosure modeled is shown in Figure 5-1. The modeled enclosure has top and bottom surfaces that are isothermal, while the side walls are adiabatic. The kinematic boundary conditions are no-slip at all surfaces except for the top, stress-free plane. All modeling in this study is conducted for liquid tin, the important physical properties of are given in Table 5-2.

Table 5-2. Properties of liquid tin [Smi67].

Property	Symbol	Value	Units
Density	ρ_o	6.713	g/cm^3
Thermal Expansion Coefficient	β	26×10^{-6}	K^{-1}
Kinematic Viscosity	ν	1.64×10^{-3}	cm^2/s
Thermal Diffusivity	κ	0.203	cm^2/s
Prandtl Number	Pr	0.008	-(unitless)-

5.4 Determination of Flow Birfurcation Points

5.4.1 Determination of Ra_{c1}

Although the first critical Rayleigh number is independent of Prandtl number and has been characterized by several authors, values of Ra_{c1} obtained during the course of this study are, nonetheless, presented here for two reasons:

1. To verify the algorithm against previous Ra_{c1} work.
2. To ensure the convergence of steady flow fields. The second critical Rayleigh number results that are presented in this work (which do depend on Prandtl number and have not yet been characterized) must use the converged steady flow profiles as their initial guesses, as fluid history can affect computed results [Ups83]; therefore, the profiles that are used are relevant in gauging the accuracy of the results.

In this study, the precise values of the first critical Rayleigh number can be determined accurately by calculating the surface-averaged Nusselt number, Nu_s. Defined as

$$Nu_S = \frac{1}{\gamma_1 \gamma_2} \iint_{x,y} \frac{\partial T}{\partial z}\Big|_{z=0,1} dx\,dy, \qquad (5\text{-}21)$$

(where γ are the aspect ratios) the surface-averaged Nusselt number is the ratio of conductive heat transfer to all heat transfer. Thus, as $Nu_s \Rightarrow 1$, $Ra \Rightarrow Ra_{c1}$, because all heat transfer becomes conduction (i.e., convection disappears). For a given aspect ratio, the velocity and temperature profiles are computed for a given Rayleigh number. The surface-averaged Nusselt number is calculated for that result. After several calculations are performed, the Rayleigh number at which Nu_s extrapolates to unity is taken as Ra_{c1}.

5.4.2 Determination of Ra_{c2}

A method analogous to the surface-averaged Nusselt number is not immediately apparent for determining Ra_{c2}. For this study, the second critical Rayleigh number was isolated by noting the velocity components and dimensionless temperature at a preselected point to see if periodic oscillations develop. If, for that Rayleigh number, no oscillations are detected, its velocity and temperature results are used as initial guesses for the next calculation, with an increased Rayleigh number. This process is repeated until oscillations are detected, and the lowest Rayleigh number at which oscillations are detected is taken as the second critical Rayleigh number.

The position of the point at which the oscillations were measured was strategically selected to be close to where the surface of a crystal would be if an actual crystal growth experiment were being conducted. The convection that is closest to the surface of a growing crystal would tend to diminish the compositional and morphological homogeneities more than convection elsewhere in the compositional domain. The point was selected to be in the center of the X-Y plane and at the grid point closest to 10% of the Z-distance from the bottom plane. To ensure the independence of the calculated results to the placement of this point, several calculations were performed with this point elsewhere in the domain, and no change was detected in the computed oscillatory transition. Thus, the Ra_{c2} predictions made by the algorithm in this work are critical events that affect the entire fluid. This supports the accuracy of this work's algorithm in predicting Ra_{c2} in addition to Ra_{c1}. (Later, a mathematical test is presented to lend more support to the accuracy of the results).

Another procedure for isolating Ra_{c2} was attempted that analyzed the amplitudes of the oscillations. For this procedure, once oscillations are detected (presumably at the second critical Rayleigh number), an additional 3-4 calculations were conducted for $Ra>Ra_{c2}$. The amplitude of the velocity components and temperature was calculated using a Fast Fourier Transform, and was extrapolated to zero. The Rayleigh number at which the amplitudes extrapolated to zero was taken as Ra_{c2}. This procedure was used for two aspect ratios; however, this analysis did not change the value of Ra_{c2} more than a few percent from what was calculated in the first approximation, thus this amplitude extrapolation method was not used further.

For all calculations, a staggered grid structure was used which placed more data points along the sides and corners of the enclosure and less in the interior. This was done to ensure resolution of side and corner effects which are extremely important in Cartesian studies. To do this, the following equations were used to form the grid:

$$X_i = X_{i-1} + \Omega \sin\left(\frac{i\pi}{N+1}\right),$$

where X indicates the grid point, N is the number of grid points in the appropriate direction, the subscript i indicates the desired grid point and i-1 is the location of the previous grid point. The function Ω is defined as

$$\Omega = \frac{L}{\sum_{i=1}^{N} \sin\left(\frac{i\pi}{N+1}\right)}.$$

5.5 Results

The results obtained for Ra_{c1} and Ra_{c2} are shown in the stability diagram in Figure 5-2. In this figure, the trend of the computed Ra_{c1} values agrees well with the linear stability results of [Cha61] for an infinite aspect ratio and [Cat72] for intermediate aspect ratios. The Ra_{c1} results also compare favorably with the numerical result of [Ozo76]. The computed Ra_{c2} values also compare well with other calculations performed for more limited problems. The Ra_{c2} trend of this work, for example, is consistent with the two-dimensional result of [Ozo95] for γ=3 and γ=4 (for Pr=0.01) and the three-dimensional work of [Nak98] for γ=5 and Pr=0.1.

The Ra_{c2} results from this study can also be checked against the theoretical predictions of Busse [Bus72] who predicts that as Pr→0, the frequency of oscillations are related to the Rayleigh number through

$$f = C\left(\frac{Ra - Ra_{c_1}}{Ra_{c_1}}\right)^{\frac{1}{2}}, \tag{5-22}$$

where C is a constant. The oscillation frequencies obtained with a Fast Fourier Transform for the different aspect ratios from this study are analyzed with a least squares regression and the result is shown in Figure 5-3. The regression analysis produces a linear fit with an R^2 value of 0.97. From the slope of Figure 5-3, a value of C=0.0303 Hz is obtained, which compares favorably with the experimental results of [Mis99] in which C=0.022 Hz is obtained.

As seen in Figure 5-2, the second critical Rayleigh number is not as sensitive to aspect ratio as is Ra_{c1}. A simple power law regression of the results indicates that $Ra_{c1} \sim \gamma^{-2.3}$ while $Ra_{c2} \sim \gamma^{-0.9}$. The difference can be explained by understanding the fundamental difference of the physics of the two transition points. When the fluid goes from no flow to steady flow, the inertia of the entire fluid must be overcome (by gravity) for convection to occur. Gravity is a <u>body</u> force, thus, the dependence on the physical arrangement to gravity is important. On the other hand, the transition from flow to oscillating flow is much different, as the resistance of the fluid to flow has already been overcome. For a fluid to begin oscillating requires only a change of motion in a fluid that is already moving. This has little to do with geometry and, thus, the dependency of Ra_{c2} on γ is less.

In the sections that follow, specific conclusions are made for each aspect ratio individually.

5.5.1 γ=0.25

Simulations carried out for γ=0.25 in the supercritical Rayleigh number regime captured some of the rich dynamics of oscillatory flow in low-Prandtl-number fluids. Using a $20 \times 20 \times 30$ grid, oscillatory convection starts at a Rayleigh number (Ra_{c2}) of 250,000, which is relatively close to the first critical Rayleigh number (Ra_{c1}=200,000), a result observed experimentally by Müller [Mul84] for liquid gallium in tall cylindrical enclosures. Figure 5-4 shows the oscillatory nature of this flow for Ra=400,000[1], using a time step of 5×10^{-5} in dimensionless units.

To visualize the intricacies of the flow oscillations, Figure 5-5 shows the oscillatory patterns the fluid makes in the x-z plane over six separate time steps 0.34 s apart. In the first three frames, there is not as much change in the overall flow pattern as in the final three. The changes that do occur are manifested in the velocity magnitudes, and in the bottom-left cell which becomes slightly more circular. Additionally, the cell in the top right of the domain moves upward slightly and becomes more compact in frame (c). Also in frame (c), a large cell in the center of the flow field begins to form. In frame (d), the cell formed in the center is evident, and in (e), it dominates the flow field, with only a secondary vortex in the lower left corner. Finally, in frame (f), the center cell has

[1] A Rayleigh number greater than Ra_{c2} was chosen to amplify the oscillations slightly to be able to visualize the oscillations better on paper.

divided into two smaller cells – in the upper left and bottom right, approximately a mirror-image of frame (a). In general, the oscillations can be thought of as a series of the following steps:

1. Begin with two cells in opposite corners
2. Slowly transform the corner cells to a single central vortex that is diagonally skewed.
3. Quickly transform the center cell into two cells in the opposite corners of (1).
4. Repeat

It should be mentioned that the orthogonal (yz) plane was also similarly analyzed and flow oscillations were manifested through a single, diagonally-skewed cell whose diameter oscillates. This flow structure, however, could not be easily visualized in two dimensions because the magnitude of these oscillations were small.

The structure of the oscillations for the first quasi-steady state profile in Figure 5-5 is shown in Figure 5-6. The cellular pattern is obvious; yet it is also clear that the flow streamline is highly three-dimensional (and does not close). The flow pattern is actually more helical than it is cellular. Also, in Figure 5-7, the constant path length of a hypothetical particle is shown as a function of time in the oscillating flow. Over the course of the 1.23 s time, the streamline coils and uncoils within the flow field.

5.5.2 γ=0.4

Using the converged Ra_{c1} profiles for this aspect ratio and a 20 × 20 × 30 grid, a Ra_{c2} value of 130,000 was obtained and the resulting velocity and temperature oscillations for this Rayleigh number are shown in Figure 5-8. As seen in this figure, the behavior of the velocity components is much more complex than those for γ=0.25, while the temperature oscillations are much simpler. For γ=0.25, the x-component of velocity (*i.e.*, u) dominated the oscillations, while for γ=0.4, the y-component (*i.e.*, v) is greater.

A macroscopic flow pattern of these oscillations was constructed and is shown in Figure 5-9. Both the magnitude of the flow velocity and the translation of the convective cells in the domain occurred over shorter time scales than for γ=0.4 than for γ=0.25. For this reason, the individual frames in Figure 5-9 are for a finer time spacing, 0.06 s. The flow begins in (a) with two nearly circular cells in the bottom right and upper left, with a smaller, cigar-shaped cell in the bottom left. There is a larger cell, diagonally-skewed in near the center of the domain. As frame (b) is reached, the lower right cell has shrunk, while the lower left has grown, although their circular and cigar shape has been preserved. The formation of the center cell has completed and is now vertical in the domain. The upper left cell has shrunk and the beginning of an upper right cell is evident. In frame (c), the upper right cell formation has completed, and on the lower left boundary there is a large cell with a much smaller vortex on the right. Frames (d), (e), and (f) are practically mirror images of (a), (b), and (c). Similar to the results shown in Figure 5-5, this oscillatory pattern can be explained overall as:

1. Begin with two large circular cells in opposite corners and two small cigar-shaped cells in the opposite corners.
2. Transfer momentum to a large, circular cell centered in the domain.
3. Transfer momentum such that the mirror image of the configuration in (1) is obtained.
4. Repeat

The results obtained from γ=0.25 and γ=0.4 suggest flow oscillations for these geometries manifest themselves in low Prandtl-number fluids in the same qualitative pattern – by the transfer of momentum from corner vortices to a centered cells to vortices in the opposite the corners. These oscillations through translation of convection cells are precisely the result expected for Prandtl number fluids [Neu90]. Had this been a high Prandtl number fluid, for example, the flow would have oscillated by periodically changing the velocity direction and/or magnitude or by changing the plane of symmetry of the oscillations.

5.5.3 γ=1.0

The second critical Rayleigh number for an enclosure with an aspect ratio of 1.0 and a $25 \times 25 \times 25$ grid, was determined to be 83,500. A trace of the velocity components and dimensionless temperature for Ra=85,000 is shown in Figure 5-10, showing definite periodicity. When analyzing the macroscopic nature of this flow configuration, changes of the cellular structure were confined to the corners of the domain and oscillate between a circular and a cigar-shaped configuration. Unfortunately, this effect is very subtle and cannot be easily visualized in a manner similar to Figure 5-5 or 5-9.

Although a macroscopic analysis of the flow oscillations was not possible for this aspect ratio, an interesting phenomenon was observed when the Rayleigh number is increased even further beyond Ra_{c2} that was not observed for the smaller aspect ratio enclosures. If, for γ=1.0, the Rayleigh number is increased beyond Ra_{c2} (as seen in Figures 5-11a through 5-11d), a secondary transition is observed beginning at Ra=100,000. In addition to the shape change, the fundamental frequency of the oscillations shifts as indicated in the Fast Fourier Transform in Figure 5-12. The frequency shift between Ra=85,000 and Ra=100,000 is more or less a constant value of 0.036 Hz for all velocity components. As far as the authors are aware, this secondary oscillation has not been observed in previous experimental or computational studies.

5.5.4 γ=2.0

Using the steady-state velocity and temperature profiles for the Ra_{c1} previously determined to be 1,800, and with a $30 \times 30 \times 20$ grid, oscillations for this aspect ratio began at a Ra_{c2} of 30,000. Figure 5-13 shows the oscillations that were detected; however, these oscillations, unlike those of lower aspect ratios, were not periodic. Several modifications to the relaxation parameters of the algorithm were made, but the same result was obtained. Similar nonperiodic oscillations were observed computationally by Ozoe and Hara [Ozo95] for two-dimensional calculations.

5.5.5 Analysis of Velocity Components

Even more can be learned about the mechanisms of the oscillations which occur in low Prandtl-number fluids through analyzing the aspect ratio effects of the velocity components and the temperature. Figures 5-14 through 5-16 show the x-, y-, and z-components for γ=0.25, 0.4, and 1.0 along with a spectral decomposition of the oscillations using the Fast Fourier Transform algorithm of the Matlab mathematics package. Figure 5-17 shows a similar analysis for the dimensionless temperatures for the three aspect ratios. An analysis of these components along with their power spectra is now analyzed individually.

5.5.5.1 X-Components

Figure 5-14 shows the x-components of velocity for the three aspect ratios studied along with the corresponding spectral decompositions. Looking initially at the velocity oscillations, an important conclusion to be drawn is that the γ=0.25 case requires nearly 30 seconds for the oscillatory patterns to develop fully, while approximately 10 s are needed for the γ=0.4 and 1.0 geometries. As the aspect ratio increases, the superharmonic increases in intensity, and for γ=1.0, several other peaks can be seen.

Probably the most significant result that can be made from the x-components is the extremely small x-component that results for γ=1.0. These velocities are of the order of 0.01 cm/s, compared with 1 cm/s for the other aspect ratios. Even more interesting is the fact that the y and z components for this aspect ratio (Figure 5-15 and 5-16) are of order 1. Since the x-component is nearly 2 orders of magnitude smaller than either the y or z components, this suggests that for γ=1.0, the two-dimensional approximation for oscillatory convection (i.e., the work of [Ozo95]) may indeed be valid.

5.5.5.2 Y-Components

The y-components of velocity in Figure 5-15 show many of the same characteristics as the x-components. Initially, the oscillations for γ=0.25 show about 20 seconds to develop, while the others about 10 seconds are needed. The transition from γ=0.25 to γ=0.4 shows the smaller peak for γ=0.25 becoming smaller for γ=0.4, while the larger peaks become markedly larger for γ=0.4. This transition can be seen in the spectral decompositions by the loss of the subharmonic in γ=0.25 and the increasing of the superharmonic in the γ=0.4 case.

Although the magnitude of the y-component for γ=1.0 is much larger than the x-components and is comparable to the other aspect ratios, the amplitude of the oscillations are quite small ($\approx$0.03 cm/s). This combined with the nearly zero x-component explains why this aspect ratio could not be visualized on paper.

5.5.5.3 Z-Components

The z-component of velocity for γ=0.25 is shown in Figure 5-16 and shows the most complicated oscillatory pattern of all the velocity components. There are two superharmonics, one of which is obvious in the spectral decomposition. The superharmonics are manifested by two smaller peaks before the main peak (one is very small). The oscillations for γ=0.4 are very similar to the mirror-image of the x-component. Finally, the z-component for γ=1.0 shows a very simple pattern.

5.5.5.4 Dimensionless Temperature

The oscillations for the dimensionless temperature for γ=0.25 are more complicated than for other aspect ratios as shown in Figure 5-17. A strong subharmonic is evident in the spectral decomposition, whose amplitude is almost as large as the fundamental frequency. The oscillations for larger aspect ratios are much simpler, although there is a small superharmonic in the amplitude for γ=1.0 that is not resolved with γ=0.4.

5.5.6 Temperature Oscillations

When the amplitudes of the temperature oscillations for the cases examined in this work are analyzed, 0.03 K, 0.2 K, and 0.28 K are computed for γ=0.25 (Figure 5-4(b)), γ=0.4 (Figure 5-8), and γ=1.0 and Ra=85,000 (Figure 5-10(d)) respectively. All of these amplitudes are much too small to be able to resolve experimentally. The oscillating range for γ=1.0 and Ra=100,000 (Figure 5-11(d)) is only marginally resolvable with an amplitude of 1.5 K. Therefore, for liquid tin, it is concluded that temperature measurements are not a useful method for determining oscillating flow patterns in very low Prandtl number fluids in agreement with the discussion of Section 1.3.

5.6. Algorithm Effects

When attempting to determine second critical Rayleigh numbers for low Prandtl-number fluids, several algorithm-dependent parameters can influence the calculated results, and their effects are a significant issue for developers of CFD codes. In this section, two of these parameters are examined in more detail – the mesh size and the Rayleigh number step size.

5.6.1 Mesh Sizing

The effects of the grid on the computed second critical Rayleigh number are one of the most significant issues facing numerical studies. Many authors have determined that calculated Ra_{c2} values can be influenced by the grid sizing used in the calculations (for example, [Ozo95, Nak98]). For the calculations presented in this work, the grid structure for Ra_{c2} calculations was the same as that used for the steady computations to eliminate the possibility of the fluid history distorting the results [Ups83]. Finer grids than those used could not be implemented because of computer memory limitations. When characterizing Ra_{c1}, however, some testing of the grid-independence was conducted for the γ=1.0 case. The original 25×25×25 mesh was lowered to 15×15×15. Both instances produced the same single-celled pattern, although some three-dimensional aspects of the flow, such as corner vortices, were not resolved to the extent as the finer grid.

5.6.2 Rayleigh Number Step Size

Another algorithm-dependant parameter that was qualitatively analyzed was the step size of the Rayleigh number used for obtaining Ra_{c2}. This was analyzed specifically for γ=1.0 enclosures. Beginning from a Ra_{c1} of 3,800, the Rayleigh number was increased to 20,000 and the velocity components and temperature were traced over a period of time to see if oscillations occur. For γ=1.0 and Ra=20,000, no oscillations were detected; therefore, the Rayleigh number was increased to 40,000 (i.e., a Rayleigh number step-size of 20,000). Again, no oscillations were detected. The Rayleigh number was increased by 20,000 until oscillations were detected at Ra=100,000. From this point, different Rayleigh numbers were used until Ra_{c2}=83,500 was isolated.

The choice to use a Rayleigh number step sizing of 20,000 was made arbitrarily. Therefore, to test the robustness of the results to this effect on the calculated Ra_{c2}, the step size was reduced to 5,000 (again, an arbitrary value). That is, from Ra_{c1}=1,800, calculations were conducted for Rayleigh numbers of 5,000, 10,000, 15,000, etc. In this series of calculations, no detectable oscillations were found for Rayleigh number up to even 150,000. With this unexpected result, the Rayleigh number was increased to 200,000 (a value which definitely should be in the oscillatory regime); still no oscillations were detected. Being even more puzzled, the Rayleigh number was increased to the extremely high value of 500,000. At this point, chaotic flow was detected, corresponding to the presence of the Ra_{c3} regime, which again is expected. This series of calculations cast some doubt on the accuracy of the Ra_{c2}=83,500 result. Therefore, the Rayleigh number step was increased to 25,000. After performing these calculations, the value of Ra_{c2}=83,500 was determined and is close to the previous value with a 20,000 step size.

This result suggests that if oscillatory calculations are performed with a Rayleigh number step size that is "too small," the algorithm may artificially suppress oscillations. When the Rayleigh number step size is too small, there is not enough change in the flow and temperature fields to cause the algorithm to converge to a new solution of the conservation equations and no new flow structure (or flow regime) is observed. This result is significant because it presents to the researcher a trade-off. On one hand, several intermediate computations between Ra_{c1} and Ra_{c2} are necessary because: (1) Ra_{c2} is usually not known *prima facie* and the intermediate calculations are successive guesses; and (2) The dynamic states of any fluid (and especially low Prandtl-number fluids) depend on the past history of the fluid [Ups83]; therefore different intermediate flow fields might very well predict different (and presumably incorrect) flow configurations. Subsequently, oscillatory flow predictions, which subsume field behavior at lower Rayleigh numbers, should be more accurate. Yet on the other hand, as this work demonstrates, too many intermediate steps suppress oscillations altogether. When considering the Rayleigh number step size there is, therefore, a balance researchers must strike: (1) The step size must be small enough to ensure the fluid history is accounted for; and (2) The step size must not be too small to avoid the masking of the Ra_{c2} transition.

5.7 Tilt

All previous calculations have assumed the z-axis of the enclosure to be parallel with the gravity vector – an arrangement that makes computations easier yet is extremely difficult to obtain in practice. If an experimental apparatus is tilted in such a way that the x and/or y axis has a gravity component, the flow dynamics are markedly different; therefore, any attempt to characterize natural convection should account for the possibility of tilt.

For the numerical results that are presented in this section, the gravity vector (or the Rayleigh number) is tilted so as to have components in multiple directions. The results are shown in Figure 5-18, which compares the stability diagram for different tilting angels with that for no tilt [Gup98]. The presence of tilt suppresses the bifurcation of the Rayleigh number as the Nusselt number only approaches unity asymptotically. The result in Figure 5-18 is significant in that even the very modest tilt of 5° completely alters the flow dynamics. If the fluid container is tilted, the first critical Rayleigh number transition is masked and cannot be resolved. Thus, any attempt to resolve the first critical Rayleigh number transition must be diligent in controlling fluid tilt.

Many researchers have found it useful to regress relations between the Nusselt and Rayleigh numbers for everyday usage. It was decided to fit the results to the form $\ln(\mathrm{Nu}) = \mathrm{A\,Ra} + \mathrm{B}$ (similar to many of the relations in Burmeister [Bur93]), from which the following coefficients are obtained:

$$\mathrm{A} = -5.94(\pm 0.24)\times 10^{-9}\,\theta^2 + 7.64(\pm 0.31)\times 10^{-7}\,\theta + 1.17(\pm 0.05)\times 10^{-5},$$

$$\mathrm{B} = 0.0222(\pm 0.0009)e^{-0.0288\theta}$$

where θ is in degrees and ranges from 5° to 60° (but does not include 0°). This model replicated the computed results with an average error of 4%.

5.8 Conclusions

An obvious conclusion from the computations conducted in this work is that Cartesian enclosures with lower aspect ratios (i.e., a tall, "skinny" geometry) exhibit oscillatory behaviors that are easier to detect and visualize because the magnitude of the velocity oscillations are comparable with the velocity vectors. The containers with smaller aspect ratios are taller and more fluid is parallel with gravity; therefore, there is more physical distance through which the oscillatory instability moves. If the aspect ratio is larger (i.e., a short, "fat" geometry) the instability can be transmitted throughout the fluid quickly, resulting in any oscillations to have amplitudes that are smaller than for larger containers.

Another important conclusion of this work is that for all cases examined, temperature oscillations did not exhibit amplitudes that were sufficient to be exploited as a means of resolving bifurcation points. As shown in Figure 5-17, the temperature oscillations for ΔT=40 K and a fluid height of 6 mm are of the order 0.1 K. Similar results were obtained for the other cases studied; only for the case of γ=1.0 and Ra=100,000 did temperature oscillations produce amplitudes that were even marginally measurable (around 1.5 K). Finally, the heat transfer characteristics of tilted enclosures heated from below were characterized, demonstrating that even a small tilt of the container with gravity masks the first critical Rayleigh number transition.

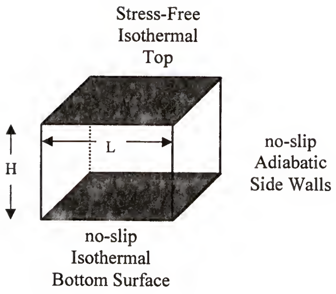


Figure 5-1 Modeled geometry and boundary conditions for oscillatory simulations in this work.

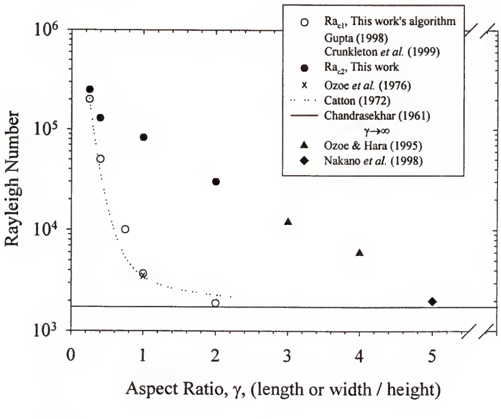


Figure 5-2. Calculated stability diagram for liquid tin (Pr=0.008).

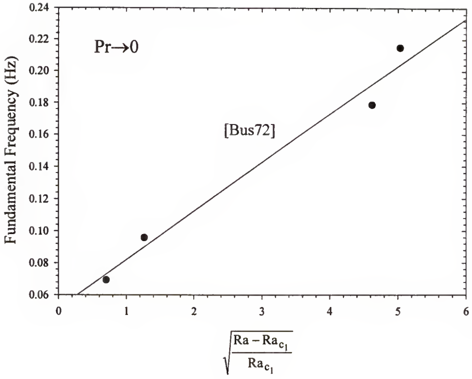
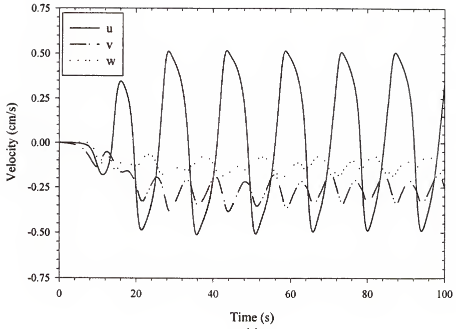
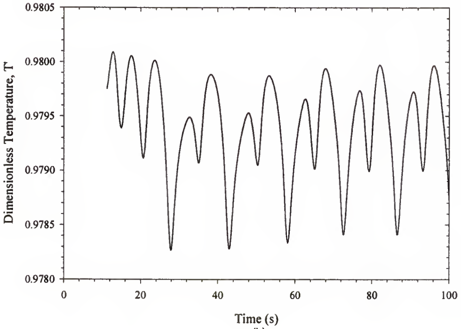


Figure 5-3 Comparison of the results of this work to the mathematical test of Busse [Bus72]. The correlation coefficient (R^2) of this linear regression is 0.95 which supports the conclusion that the Ra_{c2} results of this work are consistent with the mathematical test of [Bus72].



(a)



(b)

Figure 5-4. Periodic oscillations for Ra=250,000.and γ=0.25 taken at a point in the middle of the xy plane and at the first grid point 10% from the bottom. (a) Velocity components; (b) Dimensionless temperature

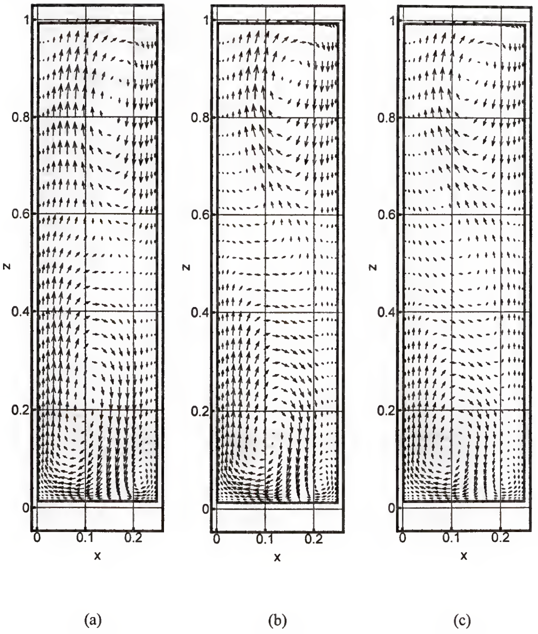
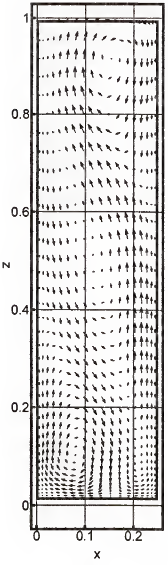
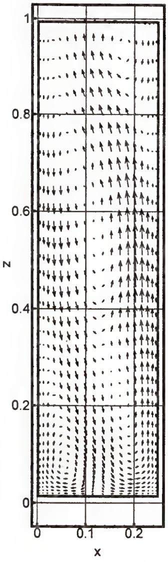


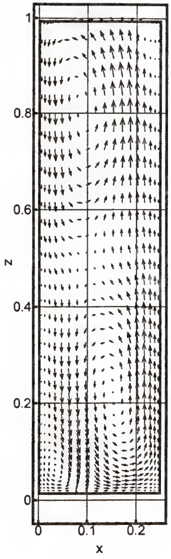
Figure 5-5. Visualization of oscillations for Ra=250,000 and γ=0.25. Each frame is 0.34s apart. The oscillations in this series of frames are manifested in the flow field by corner cells moving between the center of the domain and to the opposite corners.



(d)



(e)



(f)

Figure 5-5—continued

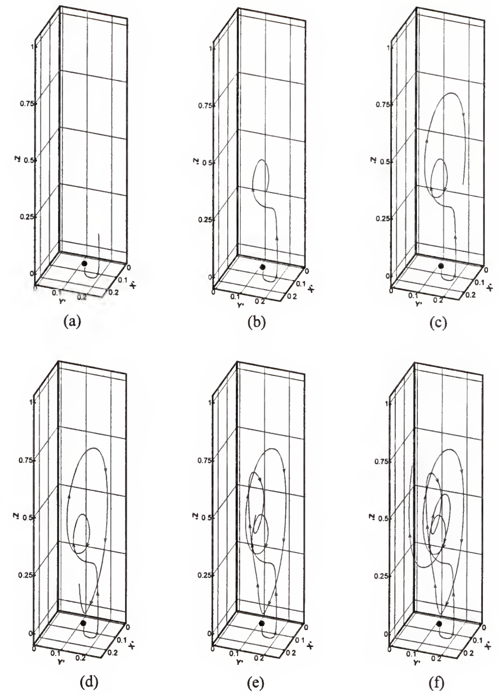
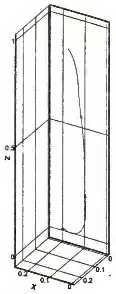
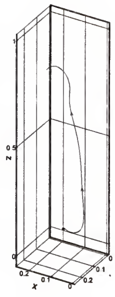


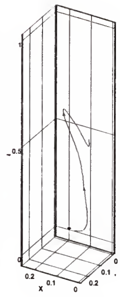
Figure 5-6 Streamline trace of quasi flow profile. Starting point is (x=0.125, y=0.125, z=0.05). Frames (a)-(f) represent increasing spatial integration steps. The helical nature of the flow structure is evident.



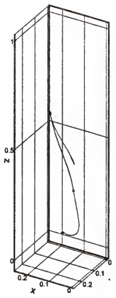
(a) t=1.17s



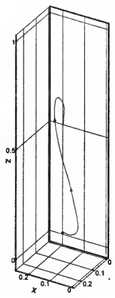
(b) t=1.23s



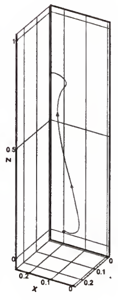
(c) t=1.30s



(d) t=1.37s

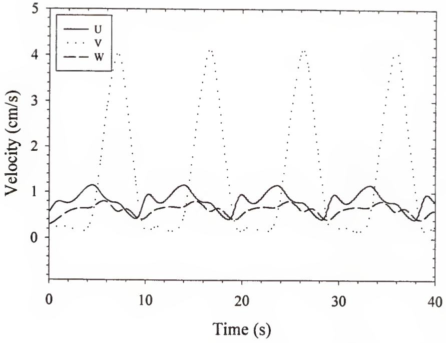


(e) t=1.44s



(f) t=1.51s

Figure 5-7 Constant-length streamline of oscillating flow for different time integration steps. The curling and uncurling of the streamline is evident (although the streamline does not close).



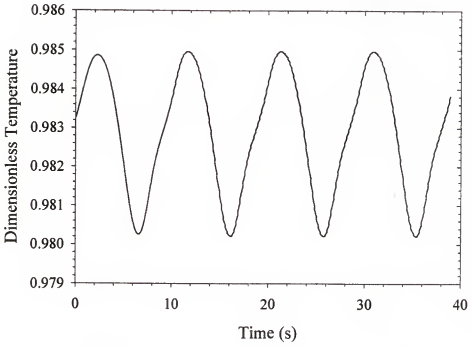


Figure 5-8 Flow oscillations for Ra=130,000, γ=0.4. Data is tracked at a point in the middle of the xy plane and at the first grid point 10% from the bottom.

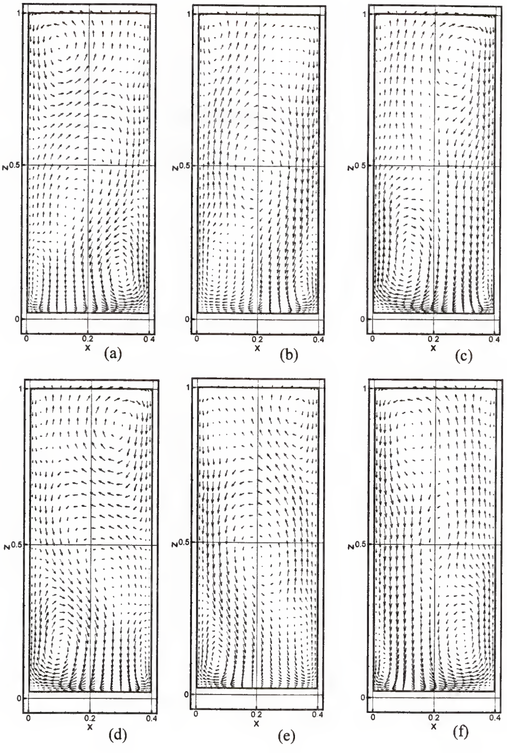
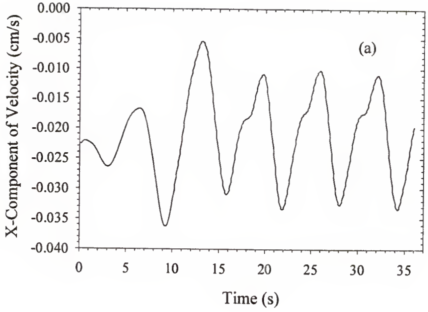


Figure 5-9 Flow visualization for Ra=130,000 and γ=0.4. Each frame is 0.06 s apart. In much the same way as with Figure 5-5, corner cells form center cells and then move into the opposite corners.



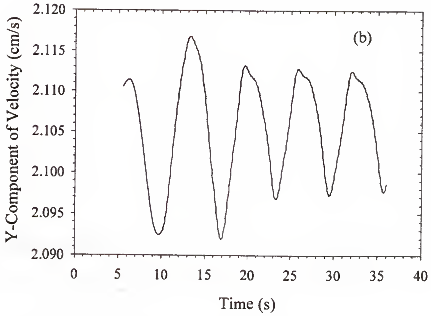
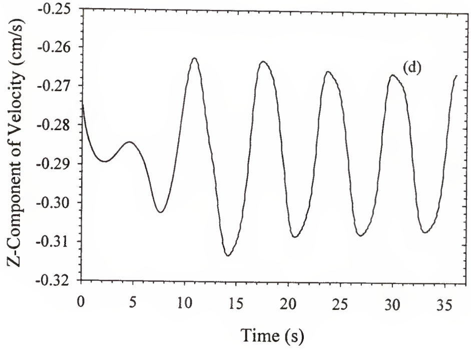


Figure 5-10 Periodic oscillations for Ra=85,000 and γ=1.0. Data is tracked at a point in the middle of the xy plane and at the first grid point 10% from the bottom. (a) X-Component; (b) Y-Component; (c) Z-Component ; (d) Dimensionless Temperature



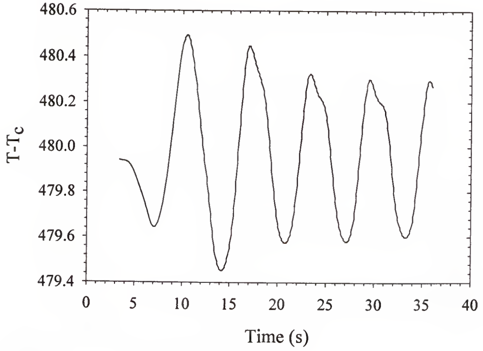
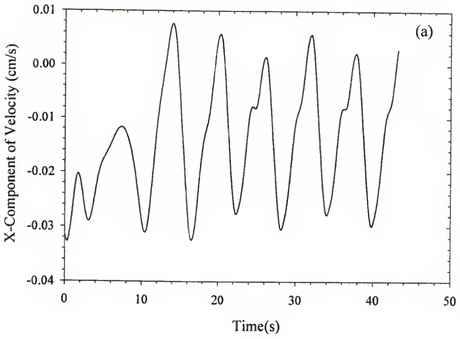


Figure 5-10—Continued



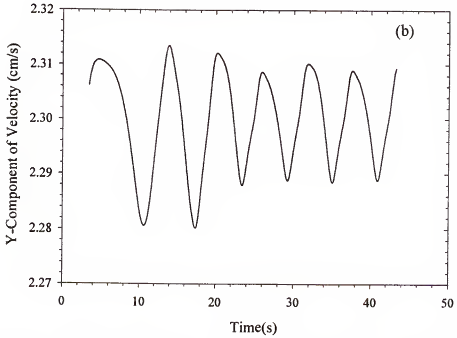
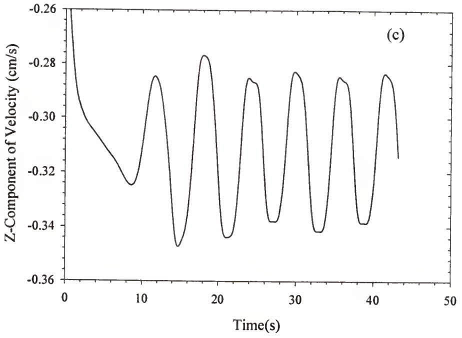


Figure 5-11 Development of secondary oscillations in Ra=100,000, γ=1.0. Data is tracked at a point in the middle of the xy plane and at the first grid point 10% from the bottom. (a) X component; (b) Y component; (c) Z component; (d) Dimensionless temperature



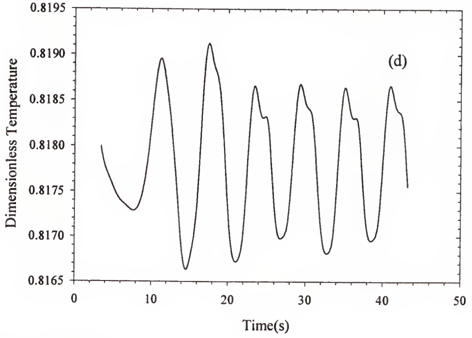


Figure 5-11—continued

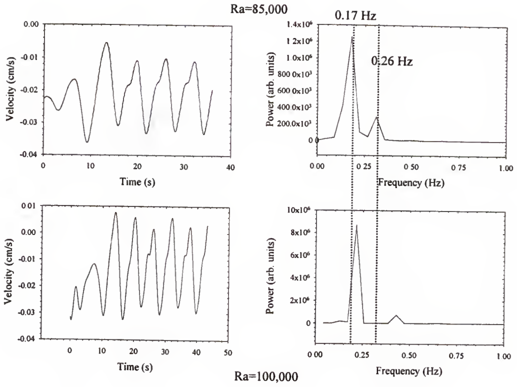
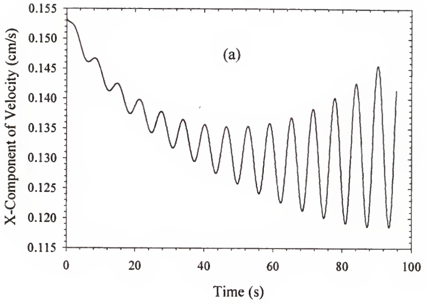


Figure 5-12. Frequency shift transition between Ra=85,000 and Ra=100,000 for γ=1.0.



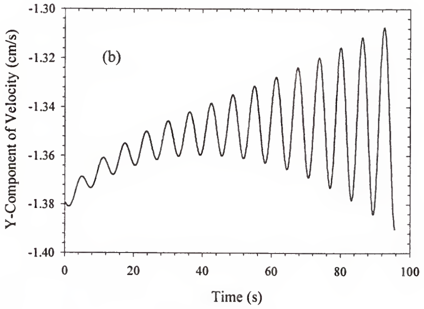


Figure 5-13 Non-Periodic oscillations for Ra=30,000 and γ=2.0. Data is tracked at a point in the middle of the xy plane and at the first grid point 10% from the bottom. (a) X component; (b) Y component; (c) Z component; (d) Dimensionless temperature

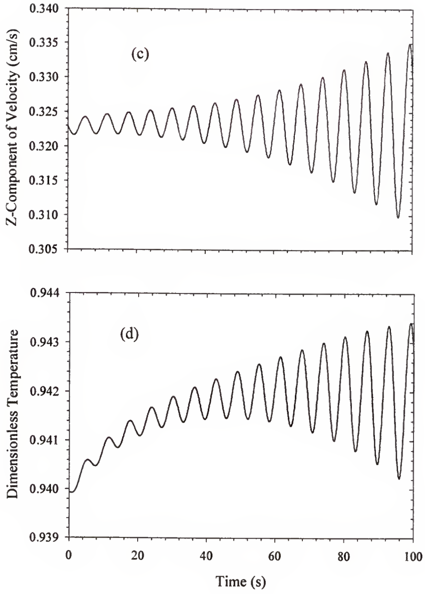


Figure 5-13—continued

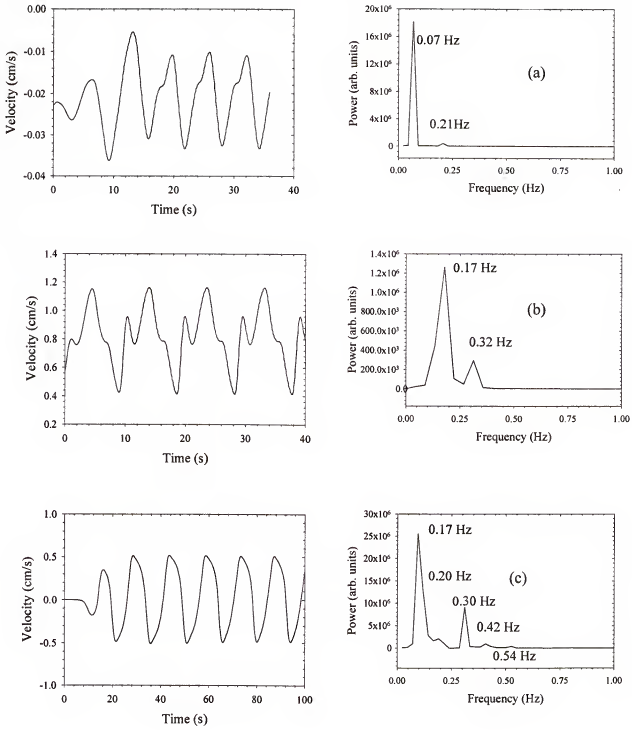


Figure 5-14. Fast Fourier Transforms of X-components of velocity. (a) $\gamma = 0.25$; (b) $\gamma = 0.4$; (c) $\gamma = 1.0$.

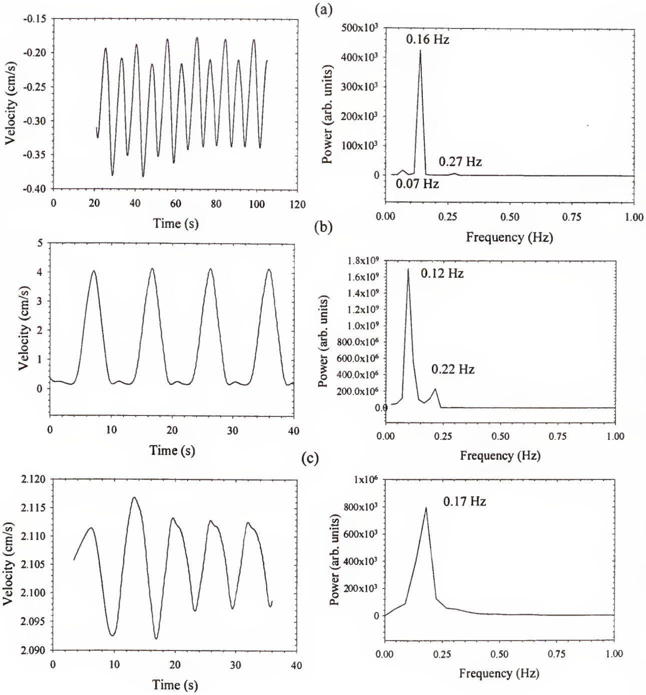


Figure 5-15. Fast Fourier Transforms of Y-components of velocity. (a) $\gamma = 0.25$; (b) $\gamma = 0.4$; (c) $\gamma = 1.0$.

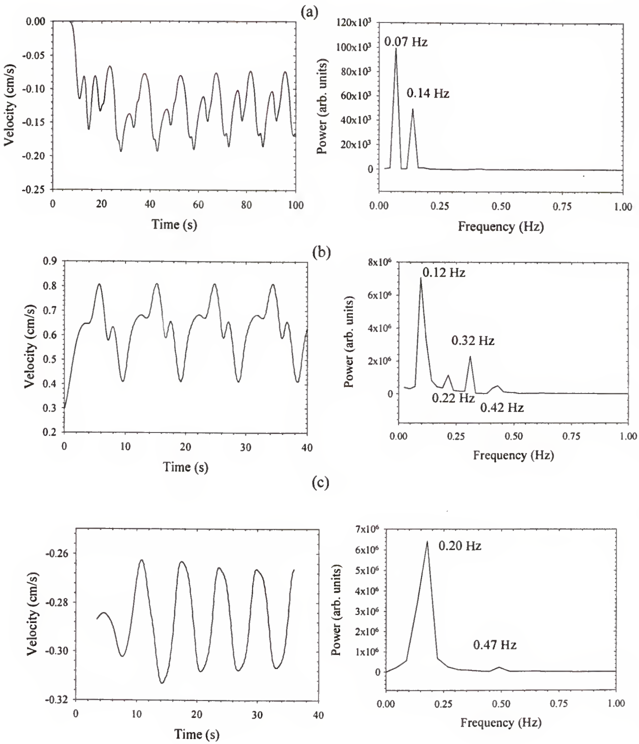


Figure 5-16 Fast Fourier Transforms for Z-components of velocity. (a) $\gamma = 0.25$; (b) $\gamma = 0.4$; (c) $\gamma = 1.0$.

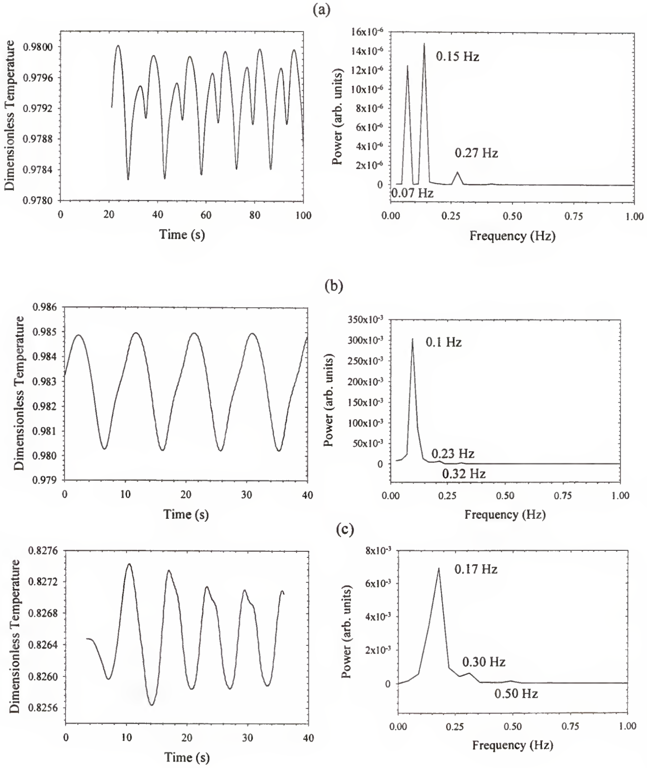


Figure 5-17. Fast Fourier Transforms of Dimensionless temperature (a) γ =0.25; (b) γ =0.4; (c) γ=1.0.

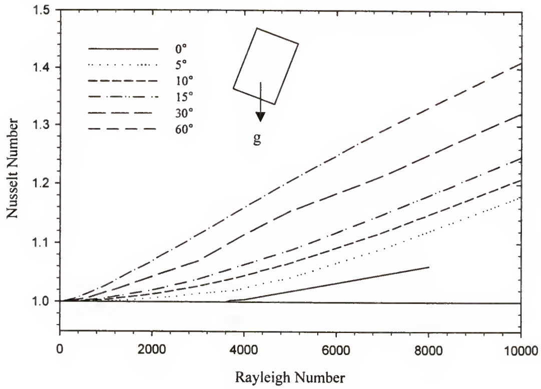


Figure 5-18. Stability diagram for three-dimensional Cartesian enclosures with tilting in one azimuth.

CHAPTER 6
FLOW VISUALIZATION USING ELECTROCHEMICAL MEASUREMENTS

6.1 Techniques for Visualizing Convection

Several experimental methods have been developed to detect the presence of convection in low Prandtl number fluids, and have been reviewed recently by Eisele *et al* [Eis98a, Eis98b]. One such method employed by several authors has been to sense temperature differences across an enclosure of fluid. For example, Moreno *et al.* [Mor90] analyzed systems of silicone oil with Prandtl numbers of 0.89 and 2.82 in cylindrical enclosures with an aspect ratio (diameter/height) of 6.63, measuring temperature at 204 thermocouple positions along the sides of the enclosure. This study predicted Ra_{c1} values of 1830 and 1770 for the two Prandtl numbers, similar to the calculated value of Charlson & Sani [Cha70] of 1723. As concluded in Chapter 5, however, visualizing convection in fluids with extremely low Prandtl-numbers (such as those in this work) is not well-suited to temperature field measurements. The fact that the Prandtl-number is low necessarily means that the fluid dissipates heat rapidly, resulting in the tendency of any temperature gradient to be dispersed quickly. Subsequently, temperature measurements are of limited use for the applications in this work.

Other authors have used suspended tracer particles to trace the streamlines of convective flows. For example, Ueda *et al.* [Ued82] employed holographic interferometry to study convection in glycerin. In this study, alimuna particles with a diameter of ~1 μm are injected into the fluid, and a 514.5 nm Ar laser is used to detect the trace patterns in the flow field. As the alumina particles move along with the flow, the streaks made by the particles' momentum are detected by the laser and are interpreted as a 2D velocity field.

Radioscopy techniques have also attempted to resolve convective patterns using tracer particles; such studies include those of Kakimoto [Kak88, Kak95] and Kostner [Der97, Kos97]. Kakimoto [Kak88, Kak95] studied flow dynamics in cylindrical enclosures of liquid silicon, while the Derebail and Koster [Der97] used X-ray sensing methods to measure the natural convection in liquid gallium ($Pr = 0.027$). Radioscopic sensing uses an X-ray source and a tracer particle. For the Kakamoto and Eguchi study [Kak88], the chosen particle was a 0.5 mm tungsten cylinder coated with ~1 mm SiO_2 and a 10 to 100 μm carbon skin.

As with temperature measurements, tracer particle techniques are not effective when viewed from the perspective of actual crystal growth processes, *e.g.*, vertical Bridgman growth. The vertical Bridgman technique is an example of a crystal growth technique in which one may need to sense the flow patterns in low Prandtl-number melts in a way that is both non-intrusive and passive. Each technique discussed above requires either: 1) a transparent fluid enclosure; or 2) the addition of a tracer particle. When crystals are grown, it is often not possible or impractical to use transparent enclosures, and the addition of tracer particles would contaminate the crystal, deteriorating the quality. The electrochemical titration technique used in this study overcomes both of these limitations by being applicable to opaque fluids and not requiring tracer particles.

In this chapter, the application of this measurement is applied to detecting convection in low Prandtl number fluids. Convection is studied in three different experimental systems:

- Vertical ampoules

- Horizontal cavities
- Multi-YSZ Flow Visualization Sensor

The hints of convection that was discovered in the diffusion measurements (Chapter 4) are examined in greater detail through EMF measurements in vertical ampoules. Second, steady current measurements in horizontal cavities are analyzed with CFD simulations. Finally, thermal convection in the flow visualization sensor (Figure 1-5) is analyzed.

6.2 Vertical Ampoules

In Section 4-3, the presence of convection in diffusion experiments was suggested by noting that the measured diffusivities for top depletion (i.e., unstable) experiments were, on average, 50 % larger than for bottom depletion (i.e., stable) experiments. This result hinted at the presence of convection in diffusion experiments that is now studied in more detail.

To study this effect in more detail, different solutal Rayleigh numbers are imposed across a fluid by applying different concentration gradients across the fluid. From the Nernst equation, the applied electrical potential corresponds to an applied concentration boundary condition; thus, different (solutal) Rayleigh numbers can be applied across the fluid by applying different potential at different points in the fluid. Because the depletion experiments are a non-steady process, the Rayleigh numbers are not as simply determined as before; thus, some explanation is warranted at this point. In a depletion experiment, the oxygen depletes from the fluid in a manner depicted in Figure 6-1. At a given time, the current at the depleting face is measured, which is related to the concentration flux, dc/dz, through Equation (2-20). The concentration flux is then used to determine the height of the fluid used in the calculation of the Rayleigh number. Thus, the Rayleigh number is a function of time because the height over which the concentration gradient is present is, itself, a function of time. The Rayleigh numbers reported in the following experiments are for the largest Rayleigh numbers obtained using this procedure because once the fluid begins to convection (indeed, if the fluid begins to convect), lowering the Rayleigh number will not stop convection – *i.e.*, once convection begins, it continues.

To study this experimentally, a series of six experiments were performed with three different depleting voltages: 0.997 V, 1.194 V, and 1.320 V for top and bottom depletion cases, corresponding to solutal Rayleigh numbers of 16,000, 56,000, and 55,000 respectively. This was done in a rectangular enclosure with width and breadth of 5 mm and a height of 6.9 mm. All other experimental details are identical with the procedures outlined in Section 4.1. All experiments began with initial EMF values within ±10 mV of one another. As with the diffusion experiments, the EMF was tracked at the opposite YSZ sensor (as explained in Section 4.2.1) and the depleting current was measured at the active face. The resulting EMF behavior for bottom and top depletion experiments is shown in Figure 6-2 and 6-3 respectively. A cursory comparison of Figures 6-2 and 6-3 indicates a dependence of the measured EMF response with pumping voltage for top depletion experiments that is not seen for bottom depletion experiments. The fact that a dependence of the diffusivity to the depletion EMF is observed for top and not bottom depletion experiments suggests the presence of solutal convection effects at higher solutal Rayleigh number experiments in the same manner as observed in Figure 4-4.

Examination of Figure 6-2 also shows the flow bifurcations of the fluid. When the diffusivity of the top depletion experiment with the lower Rayleigh number (depletion potential = 0.997 V, Ra_s=15,000) is determined through EMF measurements at the bottom cell, a diffusivity of 5.01×10^{-5} cm^2/s was measured, a value similar to what was measured in the bottom depletion (i.e., stable) experiments as well as by Equation (4-1). The experiments with higher Rayleigh numbers (55,000 and 56,000) showed predicted diffusivities that differed by nearly an order of magnitude from Equation (4-1). From the stability diagram in Figure 5-2, the first critical Rayleigh number for this configuration is around 18,000. Therefore, the discrepancy between the measured diffusivity and EMF/time behavior for Ra_s=15,000 and that of the Ra_s=55,000 and Ra_s=56,000 experiments (for only the top depletion experiments) is consistent with a flow transition of the fluid from quiescence to convection at Ra_{c1}.

Interestingly, the measured depletion currents do not show this behavior. Figures 6-4 and 6-5 show the depletion currents for the same six experiments just described. The comparison of top versus bottom depletion does show an effect of the current with Rayleigh number for the top depletion experiments, but only as an change of intercept and not of the slope. If the diffusivity is measured from the slope of the current response curve as explained in Section 2.3.2, there is only a 20% difference between the three top depletion experiments, in much better agreement than the order of magnitude difference observed for the EMF experiments.

This is an extremely important conclusion that deserves further explanation. From the comparison of the top depletion experiments measuring EMF (Figure 6-2) and current (Figure 6-4), it is seen that current measurements of the diffusivity do not seem to be as susceptible to convection influences than EMF measurements. This suggests that future researchers should investigate the possibility of conducting diffusion and convection experiments which apply a current (i.e., Neumann) boundary condition instead of the EMF (i.e., Dirichlet) boundary conditions that have been the emphasis in previous studies. The reason why this is true can be understood by simply recalling that the Rayleigh number depends on the concentration gradient across the entire enclosure, not the concentration flux at one side of the fluid. The Rayleigh number is what determines the degree of influence of convection on the flow, and the flux in or out of an electrochemical cell has nothing to do with the Rayleigh number.

In Figures 6-4 and 6-5, the drop in current from zero at 0 s (not shown on the graphs) to the first measurements made at ~ 60 s is due to the IR drop of the electrochemical cells studied. For bottom depletion experiments, the IR drop is the same for all Rayleigh numbers investigated (what is expected), but such is not the case for top depletion experiments. The fact that this is observed for only top depletion experiments is most likely due to vapor gaps that are formed between the fluid and the sensor. For bottom depletion experiments, current measurements are made at the bottom sensor and no vapor gap can be present. On the other hand, current measurements made from the top sensor (in top depletion experiments) may be so influenced. To reduce this discrepancy in the future, it should be ensured that the fluid completely fills its enclosures. This can be done by intentionally overfilling the container with the melt.

6.3 Horizontal Cavities

With diffusion experiments suggesting the presence of convection, it was decided to deliberately introduce convection into the diffusion ampoule and to measure the dynamic response. The ampoule studied had the same geometry as the one in Section 6.2; however, to ensure a convective response, this ampoule was rotated 90°. This configuration ensures convection because, as concluded in Section 5.7, tilting guarantees a convective response. Various EMF gradients were applied to the "left" and "right" sensors, corresponding to concentration gradients applied across the fluid (and different solutal Rayleigh numbers). The steady current at both sensors was then measured. For these experiments, it is imperative to ensure a uniform initial concentration profile throughout the melt before imposing the solutal gradient. In these studies, the concentration throughout the melt was brought close to equilibrium and concentration uniformity was ensured by applying the left and right concentration initial conditions until the measured currents at both active faces were constant, taking a time of at least D/H^2, where D is the diffusivity and H is the fluid height. Afterwards, the open circuit EMF was measured at the left and right faces to ensure the EMF difference was less than 10mV. For most of the low solutal Rayleigh number experiments, the EMF difference between right and left was usually about 1 to 5 mV. This process usually took 2 to 3 days. This procedure was much more rigorous than was used for vertical measurements because: (1) configurations with vertical active faces are unstable to even the smallest solutal gradients; thus, the most uniform concentration profile possible is needed. (2) the results obtained are compared with numerically computed currents, and non-uniform initial concentration fields lead to divergence in the CFD predictions (this is discussed later in this chapter).

Before getting to the results, a few words should be mentioned about the concentration gradients that were applied. To be able to compare the experiments performed to the results from CFD modeling, the experiments must be within the Boussinesq concentration range. That is, the ratio of the concentration difference to the average concentration must be as small as possible. Since EMF is related to concentration exponentially (*cf.*, Equation 2-7), the Boussinesq criterion is, in fact, rather difficult to ensure as small EMF changes lead to a large change in concentration (and, thus, the Rayleigh number). For this study, the Boussinesq requirement was taken to be satisfied if $\Delta c/\bar{c} \leq 0.5$. Physically, this criterion means that that concentration gradient must be as small as possible to insure the invariance of the physical properties to the concentration. This criterion was limited by the selectivity of the potential sources used in this study. Of course, whether or not the fluid is Boussinesq has no bearing on whether or not the fluid convects; yet, the studies presented in this fluid were limited to Boussinesq experiments so as to ensure comparison with CFD modeling. Not making this simplification would require the development of non-Boussinesq algorithms which is a goal beyond the scope of this study. Because the experiments presented were so confined, concentration gradients corresponding to Rayleigh numbers in the oscillatory regime could not be induced in the flow field. Because of this, oscillatory flows were measured using the flow visualization sensor (Figure 1-5) as detailed in the next section.

With that caveat, the steady state experiment results along with the CFD modeling are shown in Figure 6-6. The experimental data show that the currents measured were remarkably constant over the Rayleigh number range investigated. (The slight upturn at lower Rayleigh numbers results from a single outlier point). This is to be expected because all Rayleigh numbers in horizontal cavities are expected to be unstable. Moreover, all experiments investigated in this work had Rayleigh numbers lower than Ra_{c2} for this geometry (determined numerically to be greater than 10^6 by Tric *et al.* [Tri00]), thus the same steady convective regime is present as reflected in constant current measurements. To model these results, the CFD code used in Chapter 5 was used to model the experimental system, varying the Schmidt numbers (i.e., diffusivities) to find the value which best represents the data. It was found that a Schmidt number of 5 best reproduces the data, which corresponds to a diffusivity of 2×10^{-4} cm^2/s. This diffusivity value is significant because it is the same as was measured by the large Rayleigh number, top depletion experiments in Section 6.2. The results from this experiment, combined with the vertical experiments of Section 6.2 conclude that the presence of convection increases the measured diffusivity by 1 to 1.5 orders of magnitude.

6.4 Detecting Convection Using the Flow Visualization Sensor

The flow visualization sensor, shown schematically in Figure 1-5, is constructed from an alumina tube of circular cross-section 35 cm in length, with an OD of 2 cm, and an ID of 1.5 cm (Vesuvius McDanel, Beaver Falls, PA). The YSZ sensors used are cylindrically shaped and are 3 mm in diameter and 4 mm in length. The YSZ sensors are attached through the alumina tube approximately 90° from one another at distances of 1 cm, 2 cm, and 3 cm from the bottom. There is an additional sensor located in the middle of the bottom face to allow the introduction of the oxygen tracer species. The seal between the YSZ and the alumina was made by a mixture of metal oxides as described in Key *et al.* [Key96] followed by a coating of Aremco Ceramabond 571L and 571P adhesive. The open end is connected to an argon delivery system described in Section 4.1. At the beginning of the experiment, the air is evacuated to a pressure of about 150-200 mTorr and purified argon is alternatively introduced and purged by evacuation to remove residual oxygen. Argon then flows freely through the system for approximately 1 day before heating the system to the desired temperature gradient.

In addition to the above configurations, others were attempted with limited success. First, a silica tube replaced the alumina tube. Although, a better vacuum was possible (~100-150 mTorr) using the Ceramabond adhesive, this adhesive became too porous after extended periods at elevated temperature. The oxide adhesive [Key96] could not be used because it was not designed to make seals between silica glass and materials such as YSZ. Second, a square cross-section alumina tube was attempted. The problem with this configuration was the difficulty in making the seal between the square tube and the silica components of the argon delivery system. Cracks developed over extended operation periods and allowed oxygen leakage.

Although the flow visualization prototype is designed with YSZ sensors located at three heights, the two upper sensors at 2 cm and 3 cm were not used in this study because they did not provide useful information. It was decided to limit the operation of the sensor to oscillating flows, which requires larger Rayleigh numbers. The Rayleigh number required for an oscillating flow for fluid heights between 2 cm and 3 cm would require a temperature gradient that could not be supplied as the experimental system is currently configured. Thus, to ensure oscillating flow, a lower fluid height (i.e., a higher aspect ratio) was investigated that needed a lower Rayleigh number to guarantee oscillatory flow. The selected height was 1.1 cm, a height for which the 2 cm and 3 cm sensor locations would not useful.

To demonstrate the operation of the flow visualization apparatus, an experiment with the flow visualization sensor is shown in Figure 6-5 for a cylindrical alumina enclosure with height of 1.1 cm and a (thermal) Rayleigh number of 4,000. The experiment performed was a introduction experiment whose initial oxygen concentration was 7×10^{-16} mole fraction, applied to all sensors for 3 hr. The oxygen packet was introduced using an applied potential of 0.556 V, corresponding to 3×10^{-7} mole fractions. The sensor response in Figure 6-7 shows a concentration response (scaled by the initial oxygen concentration) that steadily increases with time as the oxygen tracer species follows the convection streamlines. The oxygen concentration oscillates as it increases, indicating the presence of oscillating convection.

To deduce the flow pattern from the sensor response in Figure 6-7, an inverse numerical CFD calculation should be performed using one of the many algorithms available in the literature, such as the Levenberg-Marquardt algorithm [Pre92]. Such an algorithm, however, requires several hundred data points at each azimuth [Ste99]. This would require significant modifications to the prototype design that is beyond the scope of this study. In the absence of an inverse model, it was decided, instead, to look for ways to make more qualitative statements about the structure of the flow patterns. One such procedure was to examine the amplitudes of the concentration oscillations in Figure 6-7.

As seen in Figure 6-7, the amplitudes of the concentration oscillation for all four sensors increase with time, suggesting the diffusing of the oxygen tracer as it follows the convective streamlines. Furthermore, if the linearity of the oscillatory amplitudes is examined more closely, valuable information about the structure of the convection cells can be inferred. For example, when the concentration amplitudes for the 0° and 180° sensors are fit to a linear least squares model, an R^2 value of 0.95 is obtained, while the 90° and 270° sensor oscillations produce R^2 values of 0.79 and 0.64 respectively. The deviation of the amplitude linearity in the 90° - 270° azimuth can be explained by the presence of a single convection cell acting nearer the 90° - 270° direction. Thus, in the perpendicular direction (in the plane made by the 0° - 180° sensors), a more perfect diffusive response can be observed because the oxygen can diffuse through the middle of the cell as opposed to across it. The deviation observed for the concentration amplitudes in the 0° and 180° sensors is most likely because the oscillating single cell is not perfectly aligned with the 0° and 180° sensors.

A CFD simulation of the flow profile verifies this as seen in a flow snapshot shown in Figure 6-8. The flow structure in Figure 6-8 is a single cell whose position translates in the plane shown. This computation is consistent with the observations made about the amplitude linearity.

6.5 Conclusions

Convective flows were investigated using three methods – vertical ampoules with concentration gradients and two solid state sensors, steady state current measurements under horizontal concentration gradients and two sensors, and finally, a sensor with vertical thermal gradients with four YSZ sensors. In the first technique, different depletion potentials were applied to ampoules for diffusion studies which imposed different solutal Rayleigh numbers across the melt. In these experiments, the first critical bifurcation was seen in top depletion (top heavy), but not in the bottom depletion (bottom heavy) arrangements as is expected from such experiments. The predicted Ra_{c1} for the geometry investigated was 18,000 and the results obtained were consistent with this.

The comparisons between convection applied through solutal Rayleigh numbers and with CFD modeling were made using steady state current measurements in a horizontal liquid tin system (tilted 90° from the direction of gravity). These results were modeled with a CFD code and effective diffusivities were calculated that were consistent with other convection-influenced diffusivity measurements obtained elsewhere in this work.

Finally, the robustness of the flow visualization sensor was demonstrated on an oscillating convective flow. In such systems, the concentration of the oxygen tracer species oscillated as it followed the flow pattern. A single cellular flow pattern was then inferred by noting the difference in the linearity of the concentration amplitudes in the different azimuthal planes. CFD simulations further enforced this conclusion.

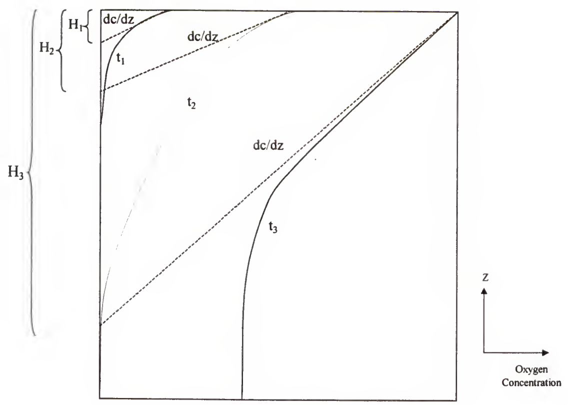


Figure 6-1 Determination of Rayleigh number for transient vertical ampoule experiments. The measured currents determine the heights over which the concentration gradient acts – Ra~H^3. (—— = successive time steps, ········· = concentration derivatives at active face).

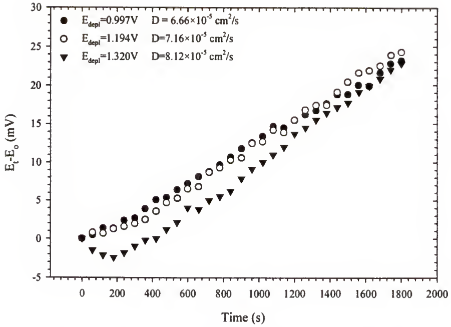


Figure 6-2 EMF response at the top cell as a function of pumping voltage (for bottom Depletion experiment). E_o=0.546 V.

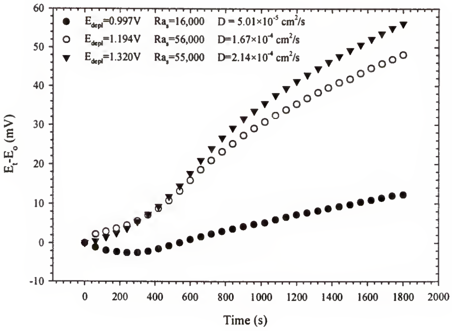


Figure 6-3 EMF response at bottom cell as a function of pumping voltage (top depletion experiment). E_o=0.542 V. The EMF jump for higher Ra_s for these and not for bottom depletion experiments suggests a convective effect.

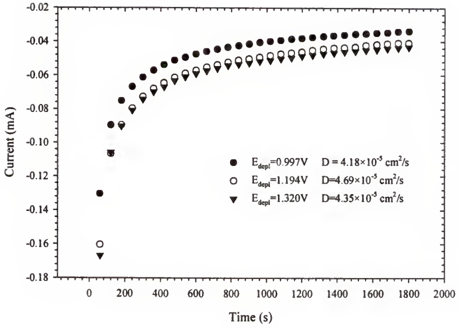


Figure 6-4 Current response at bottom cell as a function of pumping voltage (bottom depletion experiment)

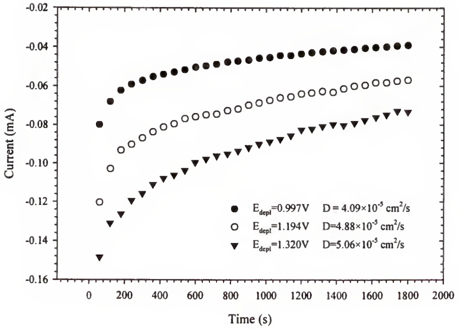


Figure 6-5 Current response at top cell as a function of pumping voltage (top depletion experiments). Note that diffusivity as measured by the slope of the current response curve is not as affected by convection as EMF measurements.

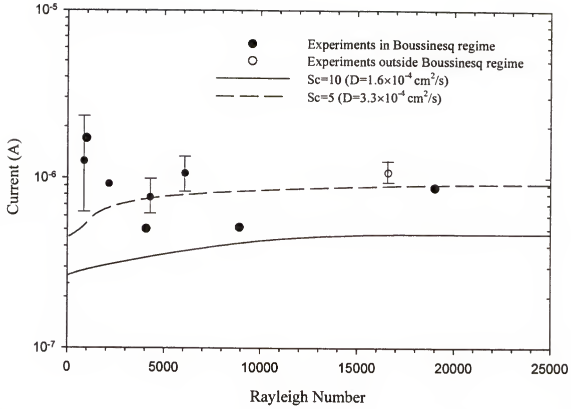


Figure 6-6 Currents measured as a function of applied solutal Rayleigh number in 90° tilted Cartesian enclosure. The lines indicate CFD simulations for a given Schmidt number. Sc=5 (D=2×10^{-4} cm^2/s) best represents the data which is consistent with other convection-influenced diffusivities measured elsewhere in this work.

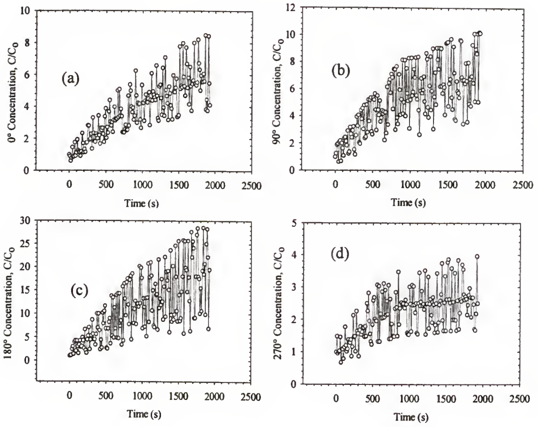


Figure 6-7 Oscillatory flow visualization experiment (a) 0°; (b) 90°; (c) 180°; (d) 270°. The data points indicate oxygen concentration measurement from that sensor. The concentration oscillations are evident.

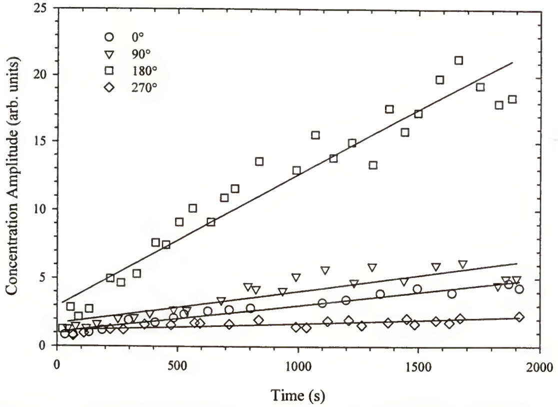


Figure 6-8 Amplitude of concentration oscillations as function of time at different sensor locations. The difference is the linearity of these amplitudes are used to infer the flow structure.

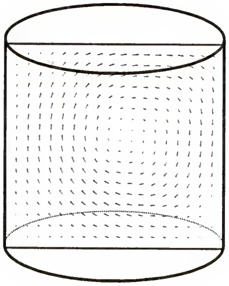


Figure 6-9. Flow snapshot for flow visualization sensor. The single cell confirms the conclusions made with the amplitude analysis in Figure 6-8.

CHAPTER 7
SUMMARY, CONCLUSIONS, AND SUGGESTIONS

The major contributions of this study have been:

- The oxygen diffusivity of liquid tin has been measured as a function of temperature over for the range of 600 K – 1200 K and represents the largest range studied.
- The oxygen diffusivity of liquid tin-lead alloys (X_{Sn}=0.1, 0.9) has been measured over the temperature range of 700 K – 1000 K. The diffusivity of the alloys was found to be lower than for the pure metal. It is concluded that additional experimentation on other intermediate alloys needs to be conducted.
- The second critical Rayleigh numbers were computed as a function of the Cartesian aspect ratio for one of the lowest Prandtl number fluids ever investigated. It is concluded that the second critical Rayleigh number is not as sensitive to aspect ratio as the first critical Rayleigh number because the transition to oscillatory convection only changes the direction of flow as opposed to overcoming the fluid resistance to convective flow.
- By comparing the difference in the measured diffusivities between unstable (*i.e.*, top depletion) and stable configurations (*i.e.*, bottom depletion), it is concluded that convection arising from either solutal gradients or tilt increases the measured diffusivity from one-half to a full order of magnitude.
- Using a fluid ampoule with a vertical applied concentration gradient, the first critical Rayleigh number was detected at a value consistent with previous modeling.
- In a cavity with horizontal concentration gradient, the measured diffusivity was confirmed to increase the measured effective diffusivity by 1 to 1.5 orders of magnitude.
- The use of the flow visualization sensor in Figure 1-5 was demonstrated by detecting oscillatory flow in a cylindrical alumina enclosure. In the absence of inverse modeling, the flow structure was inferred upon examination of the concentration amplitudes measured by the flow sensors.

The oxygen diffusivity was measured in liquid tin over the entire thermal range permitted by YSZ use. Measurements at lower temperature are not possible because YSZ is not an ionic conductor below around 580°C. Temperatures higher than those investigated in this work would be difficult because ceramic adhesives are not stable at such high temperature. Even so, the results of this study indicate the presence of convection as demonstrated by the difference in the measurements of the top vs. bottom depletion experiments. To reduce the possibility of convection even further, the following steps could be implemented:

- **Microgravity studies.** In reduced gravity, the density inversion introduced in Figure 1-1 cannot occur and buoyancy cannot build in the fluid. Thus, convection should not take place.
- **Periodic Boundary Conditions.** As suggested in Section 2.4, periodic galvanostatic or potentiostatic measurements may reduce the effects of convection in diffusivity measurements. If periodic EMF boundary conditions are imposed and combined with simultaneous current measurements, convection effects can be minimized because the concentration gradients applied across the fluid are over a much smaller height than in the experiments of this work.
- **Tilt Control.** As discussed in Sections 4.3 and 5.7, tilting of the fluid container can introduce a convective velocity that is 2-4 orders of magnitude higher than the diffusive velocity, thereby increasing the measured diffusivity. If greater effort is made to reduce (or eliminate) tilt, this effect can be minimized. This will be detailed later in this Chapter.

The alloy diffusivity measurements were limited to dilute alloy concentrations (10 mole %) because very rapid changes were expected at low concentrations; however, the difference in the measured diffusivities between 90% tin and 90% lead and their corresponding pure metals was one-half of an order of magnitude, much smaller than observed for other alloys [Hah77]. To obtain a more precise concentration functionality of the Sn-Pb system, more intermediate alloy concentration could be studied.

As discussed previously, significant refinements can be made to diffusion studies through greater tilt control. As concluded in Section 5.7, the presence of a rather insignificant 5° tilt can introduce sufficient instabilities into the fluid that will effectively mask the first critical Rayleigh number transition and guarantee convection. The current experimental configuration, for all practical purposes, has no tilt control. Currently, when constructing diffusivity ampoules, great care must be taken to ensure alignment with supporting structures; still all is a "free-handed" effort. To eliminate the effect of tilt, the following could be implemented:

- **Have the fluid ampoules professionally constructed**. Although this will greatly increase the cost of the experiments, only a few would need to be constructed to determine the degree of tilt influence on the results.
- **Redesign the furnace in a way which ensures alignment with gravity**. This can be as simple or as complex as future research demands. As an initial step, the large alumna tube containing the purge gas (into which the diffusivity ampoules sit) can be constructed of a tube with square cross section. The opening to the furnace can also be constructed from a tube of square cross section. The alumina purge tube can then fit into the furnace like a peg. Ultimately, laser guided scopes could be integrated with a control system which will maintain alignment with gravity.

It should be noted that tilt control is probably the greatest single obstacle to both the diffusion as well as the flow visualization studies. As concluded in Section 5.7, even the slightest tilt can mask the first critical Rayleigh number transition. Thus, as the flow measurement system matures, tilt control will be of utmost importance.

It is also recommended that the diffusion measurements be made with periodic, galvanic cells. Convection contamination can be reduced because the concentration gradient is applied over a smaller space in the fluid. Moreover, galvanic cells are recommended over the electrolytic one used in this study because of the Chapter 6 conclusion that current measurements seem to be less effected by the convection that is present.

For the convection modeling, there are several refinements that could increase the usefulness of the code. The code could be modified to include limited non-Boussinesq effects. Especially if future solutal convection experiments are desired, experimental results outside the Boussinesq approximation must be explained. Most of the work being undertaken today in the area of non-Boussinesq natural convection involves the thermal (or, by analogy, concentration) dependencies of the fluid viscosity (see, for example, Chang and Chen [Cha01] or Benhadji and Vasseur [Ben01]). Hypothetically, this could be integrating this into the CFD algorithm by making the Rayleigh number a position-dependant array, whose value is determined by a subroutine with a model of the viscosity temperature dependence. This model could be developed using thermodynamic and/or molecular dynamics arguments.

The application of an electrochemical flow visualization sensor to study convection in low Prandtl number fluids was demonstrated. Using an air reference electrode and a cylindrical alumina enclosure, oscillatory thermal flow was studied. The robustness of the sensor was demonstrated through its ability: (1) to detect flow oscillations; and (2) to resolve concentration amplitudes sufficiently to infer the flow pattern computed numerically.

For better comparison with modeling, several additional phenomena could be incorporated into the CFD code to add to the robustness of the predictions. The operation of the flow visualization sensor is under a thermal gradient; yet the introduction of an oxygen tracer species introduces a concentration gradient effect as well. This is the physical model of the double diffusive problem in fluid mechanics, where a fluid undergoes both a thermal and concentration gradient. (It is volunteered that this effect was completely neglected in this study). Double diffusive convection is a very complex problem whose fluid effect is only now being investigated [Col00, Ben01]. To begin to correct this, modeling work could begin on the characterization of fluid undergoing this effect. The numerical implementation would be rather simple – an additional body force term would be added rendering Equation (5-17) as

$$\frac{\partial v}{\partial t} + v \bullet \nabla v = -\nabla p + \nabla^2 v - \frac{Ra}{Pr}(T - T_o) - \frac{Ra_s}{Sc}(c - c_o).$$

Note that Boussinesq behavior is assumed for the concentration gradient. Although this would be relatively easy to code, the number of computations to be performed increases greatly because there is an additional adjustable parameter, namely the solutal Rayleigh number.

The usability of the flow visualization apparatus was demonstrated; yet, several experimental problems delayed its development for a significant period of time. The first issue is that of sensor leakage. It is very difficult to seal YSZ to alumina or silica glass at operating temperatures encountered in this study. The Aremco adhesive used in this study worked well, although experiments could only be performed for 2-3 days instead of for 2-3 weeks for the diffusion studies. To eliminate this frustration, it is recommended to perform flow visualization studies using a system similar to that used for the diffusion studies. That is, use the Cu/Cu_2O reference instead of air and enclose the apparatus in a purging tube with a flow of inert argon. Although the solid reference electrodes will operate somewhat slower than the air reference, the increased sensor reliability which will result will offset the slower operation.

APPENDIX A
CLOSE-SPACE VAPOR TRANSPORT OF INDIUM ARSENIDE ON ALUMINA SUBSTRATES

Hall devices have found important commercial applications, including Gauss meters and current sensors. An important material that is used to manufacture Hall sensors is Indium Arsenide (InAs). Its usefulness comes from its high intrinsic electron mobility. The mobility of InAs at room temperature (33,000 $cm^2/V\text{-}s$) is among the highest of any known semiconductor [Sze81]. Furthermore, the temperature dependence of the Hall potential is very low, making it suitable for operation at large temperature ranges. The methods used to grow materials for this application, therefore, have become important, considering the effect of the surface morphology, impurities, and microstructure have upon final device performance. Closed Space Vapor Transport (CSVT) is one such method and has been the topic of this research.

A.1 Closed Space Vapor Transport Processing

The Closed Space Vapor Transport (CSVT) growth process involves a "bulk" semiconductor source that is transformed into volatile species "transport agents" [Nic63]. These volatile species transport to a substrate, held at a known spacing, δ, from the source and react to form a solid film. For exothermic reactions, the temperature of the source is located at a higher temperature than the substrate. For our system, the InAs source material reacts with an $AsCl_3$ feed gas to form volatile indium chlorides and arsenic species. The temperature difference between the hot source and cool substrate produces a driving force for deposition at the cooler substrate. The deposition establishes a concentration gradient in the gas phase that gives rise to mass transfer of reactive species. It also introduces the possibility of buoyancy-driven convective transport. The significance of this buoyancy-driven flow is the main goal of this research, and is the motivation of microgravity processing.

The InAs is grown from the production of the volatile intermediates according to the following primary reactions:

$$AsCl_3(g) + \frac{3}{2}H_2(g) \leftrightarrow 3HCl(g) + \frac{1}{2}As_2(g)$$

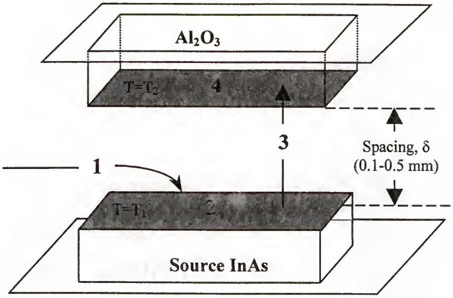
$$HCl(g) + InAs(s) \longleftrightarrow InCl(g) + \frac{1}{2}As_2(g) + \frac{1}{2}H_2(g)$$

A schematic of the CSVT process is shown in Figure A-1

In such an arrangement, Rapid Thermal Processing (RTP) is used to establish the required temperature gradient. The prototype reactor that was constructed for processing in NASA's ROMPS project is shown in Figure A-2.

A.2 Drawback to CSVT Processing: Convection

The reactions used in this InAs deposition (e.g., chlorination of InAs) must necessarily be conducted at different temperatures to provide a concentration difference between the source and the substrate. Unfortunately, when a fluid is heated unevenly so that the hotter fluid is on the bottom, a density inversion is established. This provides buoyancy to the fluid enclosure and can cause fluid motion in the reactant gases. This is the classical Rayleigh-Bénard problem in fluid mechanics. Subsequently, a study of the deposition of InAs through CSVT techniques must consider the possibility of convection. Such a study is a goal of this work.



1. Inlet Gas: 2-3% $AsCl_3$ in H_2
$2AsCl_3 + 3H_2 \dashrightarrow 6HCl + As_2$

2.Surface Reaction:
$InAs + HCl \rightarrow InCl(v) + H_2$

3. Diffusion of volatile species (InCl) to the substrate

4. Reverse of surface reaction to form a thin film of InAs

Figure A-1. Schematic of CSVT process

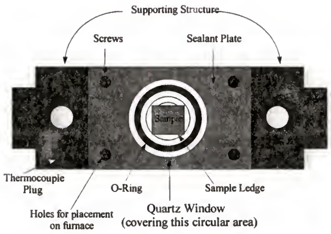


Figure A-2. RTP reactor chamber

A.3 Diffusion Modeling of InAs Deposition

In this section, the theoretical development of a model to describe the diffusion-limited growth of InAs using CSVT will be developed [Cot86]. The growth of InAs is in three distinct steps, the first of which is the production of hydrogen chloride through the reaction of a hydrogen carrier gas mixed with 3% arsenic trichloride ($AsCl_3$), represented by

$$AsCl_3(g) + \frac{3}{2}H_2(g) \leftrightarrow 3HCl(g) + \frac{1}{2}As_2(g) \; . \qquad \text{(A-1)}$$

The HCl produced by reaction (1) then reacts with the InAs source material to form volatile indium chlorides according to

$$HCl(g) + InAs(s) \longleftrightarrow InCl(g) + \frac{1}{2}As_2(g) + \frac{1}{2}H_2(g) \, . \qquad \text{(A-2)}$$

These volatile In species (along with the As species) then diffuse through the vapor gap. Ultimately, a deposition reaction can then occur at the substrate according to

$$InCl(g) + \frac{1}{2}As_2(g) + \frac{1}{2}H_2(g) \leftrightarrow InAs(s) + HCl(g) \qquad \text{(A-3)}$$

The chemical reactions given by equations (1) through (3) are those believed to dominate the process. In these equations, only indium chloride (InCl) and the arsenic dimer (As_2) are shown. In this study, the equilibrium partial pressures of several possible species that may be involved in the chemical reactions are computed and used as boundary conditions.

Once the concentrations of the chemical species are known, a diffusion-controlled growth rate model can be developed to predict the growth rate of InAs. If diffusion is assumed to dominate the process, the starting point is with Ficke's Law, which gives the molar flux for the InAs system as

$$J_{In} = D_{In,H_2} \vec{\nabla} \vec{c} = D_{In,H_2} \left(\frac{\partial c_{In}}{\partial x} + \frac{\partial c_{In}}{\partial y} + \frac{\partial c_{In}}{\partial z} \right), \qquad \text{(A-4)}$$

where D_{In,H_2} is the diffusivity of the volatile indium species in the H_2 carrier gas. If diffusion is limited to the direction perpendicular to the source and substrate (i.e., infinite extent in the radial direction), Equation (A-4) reduces to

$$J_{In} = D_{In,H_2} \frac{dc_{In}}{dx}. \qquad \text{(A-5)}$$

Assuming steady state diffusion, the concentration profile across the vapor gap is linear and can be approximated by the ideal gas law. Finally assume a constant pressure of 1 atm, throughout the entire process. These assumptions transform Equation (A-5) into

$$J_{In} = \frac{D_{In_xCl_y,H_2}}{\delta} \left[\frac{P_{In_xCl_y}(T_1)}{RT_1} - \frac{P_{In_xCl_y}(T_2)}{RT_2} \right], \qquad \text{(A-6)}$$

where δ is the vapor gap spacing and J_{In} is the total indium flux. The term In_xCl_y appears in Equation (A-6) to indicate the presence of indium flux by several volatile indium chloride species. The diffusion coefficients in Equation (A-6) are modeled by the kinetic theory of Hirschfelder [Hir54, Ske74] as

$$D_{ij} = \frac{3R\overline{T}}{8(P_i + P_j)\left(\frac{(\sigma_i + \sigma_j)}{2}\right)^2}\left(\frac{R\overline{T}}{2\pi}\frac{(M_i + M_j)}{M_i M_j}\right)^{\frac{1}{2}}. \tag{A-7}$$

Where: $\overline{T}$ = average temperature between the source and the substrate

M_i, M_j = Molecular weights of the transporting species (In_iCl_y) and the carrier gas (H_2), respectively.

σ = Molecular diameter of species i and j.

The values of σ for H_2 as a function of temperature are given by [Ant84] as

$$\sigma_{H_2} = 0.222\sqrt{\left(1 + \frac{234}{\overline{T}}\right)}. \tag{A-8}$$

The values of σ for the indium chlorides are not as readily available. Therefore, it was decided to use the logic of Anthony [Ant84], and estimate the molecular distance, σ, as 2 Å/atom. For example, Anthony [Ant84] estimates σ for Ga_2O as 8 Å. Following this, we will assume that σ for InCl is 4 Å because it has two atoms in its structure, and similar values were obtained for the other compounds.

The growth rate can then be determined by

$$GR = \frac{J_{In}M}{\rho}, \tag{A-9}$$

where J_{In} is the molar flux of indium calculated from Equation (A-6), M is the molecular weight of InAs (190 g/mol), and ρ is the density of the deposited material (5660 kg/m^3) [Tya91].

All equilibrium calculations for this work were conducted with the algorithm developed and tested by Apparicio [Apa97] at the University of Florida. For this analysis, an extensive assessment of the thermodynamic properties was conducted and the data of Gurvich and Barin were selected [Gur89, Bar95].

A.4 Diffusion Modeling

A simple equilibrium calculation was performed using the following species were present: As_2 (g), As_4 (g), $AsCl_3$ (g), H_2 (g), HCl (g). These species were chosen because they are the most abundant in the temperature range of interest. Table A-1 gives the equilibrium mole fractions of each species at various temperatures. This simple calculation indicates that the inlet $AsCl_3$ has essentially decomposed to HCl and As_2 (or As_4). The arsenic tetramer is more important at the lower temperature and the dimer at above 1500 K as driven by entropy considerations.

A second equilibrium calculation was performed with an inlet gas phase $AsCl_3/H_2$ ratio the same but with an excess of InAs (s) present and incorporating the following species: As_2 (g), $AsCl_3$ (g), Cl_2 (g), H_2 (g), HCl (g), InCl (g), InAs (s). These mole fractions are then used as a boundary condition to calculate the growth rate for a vapor gap of 0.0508 cm. With an excess of InAs(s), the equilibrium species mole fractions are reported in Table A-2 and Figure A-3 shows the predicted growth rates which turns out to be approximately linear with respect to source temperature because of the linear concentration with temperature for the species considered.

For the full calculation, additional species were added to the model, but the results were not significantly different for the HCl production reaction. The species considered for the second reaction were: As_2 (g), As_3 (g), As_4 (g), $AsCl_3$ (g), AsH (g), AsH_3 (g), Cl_2 (g), H (g), H_2 (g), HCl (g), In (g), InCl (g), $InCl_2$ (g), $InCl_3$ (g), and In_2Cl_6. The differences are primarily the inclusion of additional indium chloride species as well as other arsenic by-products. The results, again, did not significantly change with respect to temperature for most of the species.

When calculating the molar fluxes of the system, the production of the indium species is of prime importance. The temperature dependence of the partial pressure of selected In species is given in Figure A-4 for the complete reaction analysis. As can be seen, the most thermodynamically favored transporting species above 800 K is InCl(g), while over the lower temperature regime, $InCl_2$(g) and then $InCl_3$(g) are also important. Over the very low temperature regime, In_2Cl_6(g) must be included. Since there are contributions from different indium compounds, the diffusivities for each of these components must be calculated to assess their effect on the total indium flux.

From this point, molar fluxes for each of the In species was calculated for various source and substrate temperatures. As for temperature selection, the substrate temperature (T_{cool}) was assumed to be a constant 500 K as before, and the source substrate (T_{hot}) was varied in the range 500 - 1900 K. Figure A-5 shows the variation of the growth rate as a function of T_{hot}. By comparing the curves shown in Figures A-3 and A-5, it can be seen that the calculated growth rate for the simple model is significantly higher than the more complicated model. This can be understood by the fact that the In is transported by Cl and the species with higher Cl/In ratios transport less In than InCl.

Table A-1 Equilibrium mole fractions with a 3 mole % $AsCl_3$ mixture in H_2, P=1 atm, excess InAs, and T varying a shown on chart.

Temp (K)	**As_2 (g) X 10^2**	**As_4 (g) X 10^4**	**$AsCl_3$ (g) X 10^9**	**H_2 (g)**	**HCl (g) X 10^2**
500	7.35E-11	73.4	41.1	0.905	8.81
600	2.3E-08	73.4	19	0.905	8.81
700	0.00000138	73.4	11.4	0.905	8.81
800	0.0000294	73.4	8.07	0.905	8.81
900	0.000315	73.3	6.3	0.905	8.81
1000	0.00209	73.3	5.26	0.905	8.81
1100	0.00977	72.9	4.6	0.905	8.81
1200	0.035	71.6	4.14	0.904	8.81
1300	0.101	68.3	3.79	0.904	8.81
1400	0.241	61.2	3.47	0.903	8.8
1500	0.48	49.2	3.13	0.902	8.79
1600	0.794	33.4	2.74	0.901	8.78
1700	1.09	18.4	2.29	0.9	8.76
1800	1.29	8.63	1.85	0.899	8.75
1900	1.38	3.78	1.48	0.898	8.75

Table A-2 Equilibrium growth rate predictions using the reduced set of species. Constant substrate temperature of 500 K, δ=0.0508 cm.

Temp. (K)	**D_{ij} (m^2/s)**	**Molar Flux mol/(m^{6-}s)**	**Growth Rate µm/min**
500	4.31E-07	0.006634	13.34
600	5.14E-07	0.009895	19.9
700	6.02E-07	0.013244	26.64
800	6.94E-07	0.016711	33.61
900	7.90E-07	0.020268	40.77
1000	8.63E-07	0.022948	46.16
1100	9.51E-07	0.026208	52.71
1200	1.03E-06	0.029279	58.89
1300	1.13E-06	0.032781	65.93
1400	1.23E-06	0.036619	73.66
1500	1.34E-06	0.040646	81.76
1600	1.45E-06	0.044815	90.14
1700	1.57E-06	0.049105	98.77
1800	1.69E-06	0.053513	107.63
1900	1.81E-06	0.058034	116.73

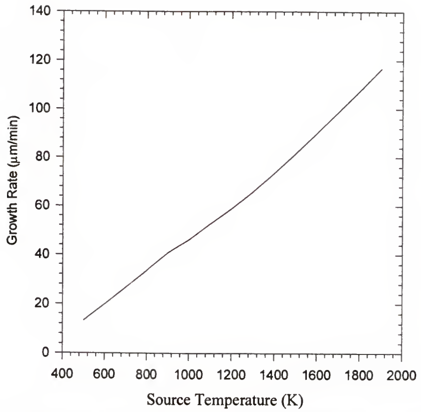


Figure A-3 Equilibrium growth rate prediction for reduced species set. Constant substrate temperature of 500 K, δ=0.0508 cm.

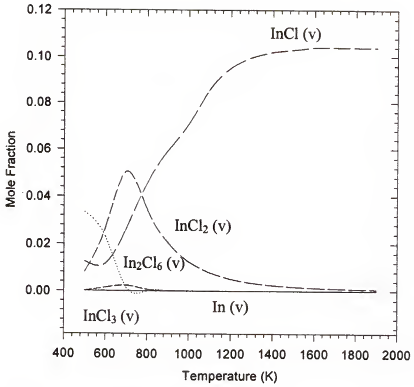


Figure A-4. Indium chloride production in expanded species set

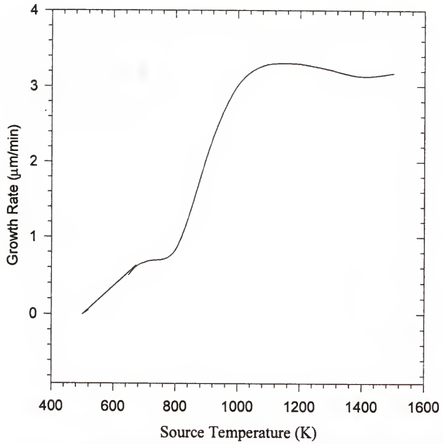


Figure A-5 Growth rate predictions for expanded species set. Constant substrate temperature of 500 K, δ=0.0508 cm.

The curve in Figure A-5 also shows two plateaus; one around 700 K and another around 800 K. This is a consequence of the transition between the regime of preponderance of InCl and $InCl_2$.

Finally, it should be stressed at this point that all previous growth rate predictions have assumed that convection is absent in the vapor gap. The presence of convection will significantly affect the rate and concentration of all transporting species, thus a discussion of the convection calculations performed is warranted at this point.

A.5 Convection Modeling

The modeling in this work sought to solve the mass, momentum, and energy balance equations simultaneously. The appropriate equations in vector form are [Bir60]

CONTINUITY $$\frac{\partial \rho}{\partial t} + \nabla \bullet (\rho \vec{v}) = 0 \qquad \text{(A-10)}$$

MOMENTUM $$\rho \frac{\partial \vec{v}}{\partial t} + \rho(\vec{v} \bullet \nabla \vec{v}) = \mu \nabla^2 \vec{v} - \nabla p + \rho \vec{g} \qquad \text{(A-11)}$$

ENERGY $$\rho C_p \frac{\partial T}{\partial t} + \rho C_p (\vec{v} \bullet \nabla T) = k \nabla^2 T\,. \qquad \text{(A-12)}$$

Initial two-dimensional calculations were performed with FSEC 1.1 (Florida Software for Environmental Calculations). The algorithm was developed by the Florida Solar Energy Council and modified significantly for use with personal computers for this project. The algorithm is a finite-element solver that solves the coupled continuity, momentum, and energy equations. During the first year of this research, simple two-dimensional velocity and temperature calculations were performed for the vapor gap spacing only. Assuming the vapor gap is composed of pure hydrogen, a temperature difference of 30 K, no-slip walls, constant temperature source and substrate, and isothermal walls elsewhere, a velocity profile was obtained showing a dual convective flow pattern such as the one shown in Figure A-6 below.

During the second year, these calculations were extended to allow the vapor in the gap to exit and flow throughout the entire reactor spacing (albeit still in a two-dimensional vector space). The velocity profile for the vapor gap in such a configuration is shown in Figure A-7.

Comparison of the results in Figures A-6 and A-7 indicate that simulations predict different flow configurations when the different geometries are modeled. Figure A-6 predicts dual convective cells inside the vapor gap; while Figure A-7 shows significantly more circulation. Figure A-7, however, is particularly interesting in that each computational node shows a flow which has a direction opposite with the next. That is indicative of a highly complex, three-dimensional flow configuration. Therefore, three-dimensional calculations were needed to reflect the true flow pattern. FSEC was unable to converge to a solution for such a configuration so during the third year of this work, these calculations were performed with a three-dimensional finite volume algorithm developed and tested by Gupta [Gup98]. This algorithm reproduced the two-dimensional flow patterns in Figure A-6 and A-7 well; however, it did not converge to a three-dimensional flow patterns for Figure A-6 (with an aspect ratio of 39). This algorithm, however, did produce a converged flow pattern for an aspect ratio of 15 (corresponding to a vapor gap of 0.1mm). This flow pattern is shown in Figure A-8.

Figure A-8 shows multiple cells aligned throughout the domain. Notice, however, that there are streamlines which are not perfect cells. From an analysis of perpendicular planes, it was determined that these are projections of cells in orthogonal azimuths.

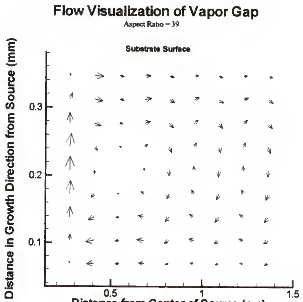


Figure A-6 2D modeling of vapor gap convection. ΔT=50 K, no-slip walls, isothermal source and substrate, isothermal side walls.

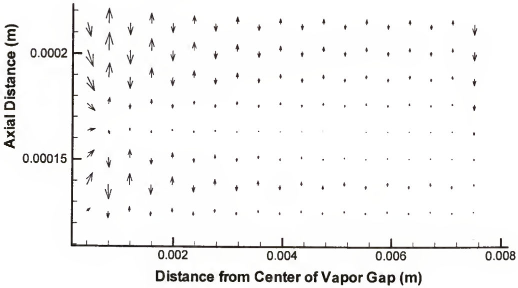


Figure A-7 2D modeling of vapor gap convection accounting for transport throughout the RTP reactor. ΔT=50 K, no-slip top and bottom, stress-free side walls, isothermal source and substrate, isothermal side walls.

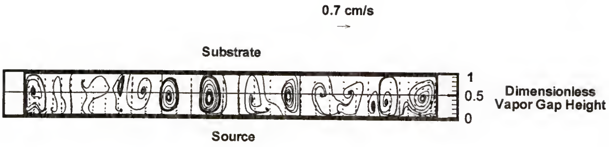


Figure A-8. Flow visualization for an aspect ratio of 15. ΔT=50K. This is a 2-D projection of a 3-D calculation. Similar results were obtained in the perpendicular azimuth.

A.6 Summary and Future Study

The major conclusions of this study include:

- The predicted growth rate of InAs using a diffusion-controlled model is very sensitive to the transport species.
- The inclusion of several possible indium chlorides as transporting agents reduces the growth rate by more than an order of magnitude.
- Convection is definitely present in the CSVT growth of InAs which use RTP. Different cellular configurations were obtained in two-dimensional calculations, indicating three-dimensional convective effects.
- Using a three-dimensional algorithm, a multicellular cellular configuration was obtained. Several cells were aligned in the diagonal directions.

Several experimental steps can be taken to reduce the effect of convection on CSVT systems. All previous experiments have used a batch reactor configuration; however, if continuous flow reactors are used (i.e., Rapid Thermal Chemical Vapor Deposition – RT-CVD), natural convection will be minimized and the quality of crystals may be improved. Such an arrangement would have the advantage of improved film quality, yet would also have the disadvantage of requiring considerably more reactant flow to grow a given film.

APPENDIX B
SOLUTION TO THE ONE-DIMENSIONAL UNSTEADY DIFFUSION EQUATION

In this appendix the series solution of the diffusion Equation (2-12) and (2-13) will be solved. The geometry and boundary conditions are shown in Figure B-1. The

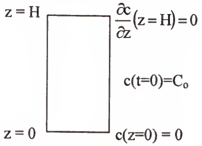


Figure B-1. Geometry of diffusion geometry

diffusion process is assumed to be axial (i.e., one-dimensional). Additionally, the diffusivity is assumed to be constant. Such a process is governed by Fick's Law, represented as

$$\frac{\partial c}{\partial z} = D\frac{\partial^2 c}{\partial z^2} \tag{B-1}$$

The boundary conditions are

$$c(z = 0) = 0 \tag{B-2}$$

$$\frac{\partial c}{\partial z}(z = H) = 0 \tag{B-3}$$

$$c(t = 0) = C_o. \tag{B-4}$$

The solution to (B-1) is assumed to take the form

$$c(t,z) = T(t)Z(z). \tag{B-5}$$

Substituting (B-5) into (B-1) gives

$$Z(z)\frac{\partial T(t)}{\partial t} = DT(t)\frac{\partial^2 Z(z)}{\partial z} \tag{B-6}$$

or,

$$\frac{1}{D}\frac{1}{T(t)}\frac{\partial T(t)}{\partial t} = \frac{1}{Z(z)}\frac{\partial^2 Z(z)}{\partial z}. \tag{B-7}$$

Since each side of Equation (B-7) are functions of a single variable, both the right and left hand sides of (B-7) must be equal to a constant. If this constant is denoted $-\lambda^2$, (B-7) can be separated into two ODE's:

$$\frac{dT(t)}{dt} + \lambda^2 DT(t) = 0 \tag{B-8}$$

and

$$\frac{d^2 Z(z)}{dz^2} + \lambda^2 Z(z) = 0, \tag{B-9}$$

the solutions to which are

$$T(t) = A^* e^{-\lambda^2 Dt} \tag{B-10}$$

and

$$Z(z) = A\sin(\lambda z) + B\cos(\lambda z). \tag{B-11}$$

Combining equations (B-3), (B-4), and (B-5) provides the boundary and initial conditions for the separated ODE, Equation (B-11). The condition at z=0 must necessarily means B=0. The condition at z=H is Z'(z=0)=0. Substituting this into (B-11) yields

$$0 = A\lambda\cos(\lambda H). \tag{B-12}$$

Neither A nor λ can be zero for a non-trivial solution, leaving only the cosine term which can be zero at values of $\left(n+\frac{1}{2}\right)\pi$. The eigenvalues, λ, are then $\lambda = \frac{\left(n+\frac{1}{2}\right)\pi}{H}$, giving a solution to Z(z) as

$$Z(z) = A\sin\left(\frac{\left(n+\frac{1}{2}\right)\pi z}{H}\right). \tag{B-13}$$

The solution, c(t,z) is the superposition of the product of (B-10) and (B-13),

$$c(t,z) = \sum_{n=1}^{\infty} A_n \sin\left(\frac{\left(n+\frac{1}{2}\right)\pi z}{H}\right)e^{-\lambda^2 Dt}, \tag{B-14}$$

The constant A_n, is obtained by applying the initial condition, (B-4), yielding

$$C_o = \sum_{n=0}^{\infty} A_n \sin\left(\frac{\left(n+\frac{1}{2}\right)\pi z}{H}\right). \tag{B-15}$$

Multiplying both sides of (B-15) by $\sin\left(\frac{\left(m+\frac{1}{2}\right)\pi z}{H}\right)$ and integrating over the diffusion length, 0<z<H yields

$$C_o \int_0^H \sin\left(\frac{\left(m+\frac{1}{2}\right)\pi z}{H}\right)dz = \sum_{n=0}^{\infty} A_n \int_0^H \sin\left(\frac{\left(m+\frac{1}{2}\right)\pi z}{H}\right)\sin\left(\frac{\left(n+\frac{1}{2}\right)\pi z}{H}\right)dz \tag{B-16}$$

The integral on the right hand side of (B-16) is zero in all cases except for n=m, when it is equal to $\frac{H}{2}$. The integral on the left hand side of (B-16) can be evaluated by substitution and A_n is

$$A_n = C_o \frac{2}{\left(n+\frac{1}{2}\right)\pi}. \tag{B-17}$$

(The index m and n are dummy index variables for summations and can be interchanged).

Substituting (B-17) into (B-14) gives the final solution,

$$\frac{c(z,t)}{c_o} = \sum_{n=1}^{\infty} \frac{\exp\left[\frac{-D}{H^2}\left(n+\frac{1}{2}\right)^2 \pi^2 t\right]}{\left(n+\frac{1}{2}\right)\frac{\pi}{2}} \sin\left(n+\frac{1}{2}\right)\pi\frac{z}{H}, \tag{B-18}$$

which is Equation (2-14).

APPENDIX C
FORTRAN SOURCE CODE FOR FLOW FIELD CALCULATIONS

The following is the FORTRAN source code used for Cartesian calculations in this work. The majority of this was written by Gupta [Gup98], but modifications in the grid computations and time stepping were made for this study. The important adjustable parameters for this code are given in Table C-1.

Table C-1. Adjustable parameters in FORTRAN source code. Values (in parentheses) indicate mass transfer case.

Code Variable	Parameter	Use
len, bre, hei	length, breadth, height	Aspect Ratio of Enclosure
Re	Rayleigh Number (Solutal Rayleigh Number)	Amount of Buoyancy from Thermal (Solutal) Gradient
Pr	Prandtl Number (Schmidt Number)	Ability to Transport Heat (Mass)
Tilt	Tilt	Tilt the Enclosure
NumTimeSteps	Number of Time Steps	Determine number of time steps
dtime	Length of time step in real time	Determine time time step

```
C     Computational Fluid Dynamic Technique of finite volumes applied to
C     a cuboid geometry  for rigid walls
C     Using equation based on kinematic viscosity for no slip BC
C     Variable Declarations

      Integer N,M,P,RC,WC
      parameter(N=30,M=30,P=25,RC=5,WC=6)
      Real Aw,Ae,An,As,At,Ab,Ap,Apo
      Real Dw,De,Ds,Dn,Dt,Db
      Real dXp,dYp,dZp,dXw,dXe,dYs,dYn,dZt,dZb
      Real Fe,Fw,Fn,Fs,Ft,Fb
      Real cu,cv,cw,ct,cud,cvd,cwd,ctd
      Real Re,Pr
      Real len,bre,hei
      Real  a,b,c,d,x
      Real dtime,times,timef
      Real cut,cvt,cwt,cutd,cvtd,cwtd
      Real m1
      parameter(len=0.87,bre=0.87,hei=1.0)
      real uerror,uerrorn,uerrord,uvwerror
      real verror,verrorn,verrord
      real werror,werrorn,werrord
      real converge,tini
      real tempc,tempavg
      Real dm1,dm2

      integer con1,con2,TimeStep,NumTimeSteps

c     Velocity  u array
      Real u(0:N,0:M+1,0:P+1),uold(0:N,0:M+1,0:P+1)
c     Velocity  v array
      Real v(0:N+1,0:M,0:P+1),vold(0:N+1,0:M,0:P+1)
c     Velocity w array
      Real w(0:N+1,0:M+1,0:P),wold(0:N+1,0:M+1,0:P)

c     initial velocity array
      Real uo(0:N,0:M+1,0:P+1),vo(0:N+1,0:M,0:P+1),wo(0:N+1,0:M+1,0:P)
      Real uinitial(0:N,0:M+1,0:P+1),vinitial(0:N+1,0:M,0:P+1)
      Real winitial(0:N+1,0:M+1,0:P)

C     b for continuity eqn
      Real barray(0:N-1,0:M-1,0:P-1)
      Real Apu(0:N-1,0:M-1,0:P-1)
      Real Apv(0:N-1,0:M-1,0:P-1)
      Real Apw(0:N-1,0:M-1,0:P-1)
      common/thomas/a(0:40),b(0:40),c(0:40),d(0:40),x(1:41)
      real xcord,ycord,zcord
      real urelax,vrelax,wrelax,Tilt,PI,KinVisc
      integer uvwtime,uvwtmax
      common/grid/Xcord(0:40),Ycord(0:40),Zcord(0:40)
      Real temp(0:N+1,0:M+1,0:P+1),tempo(0:N+1,0:M+1,0:P+1)
      Real Press(0:N-1,0:M-1,0:P-1)
c     Main program
            PI=ACOS(-1.0)
      uerror=1.
      verror=1.
      werror=1.
            KinVisc=0.00164
      Re=5000.0
      Pr=50.0
            Tilt=95.0
            Tilt=Tilt*PI/180
      dm1=1.0
      dm2=1.0/Pr
      urelax=0.1
      vrelax=0.1
      wrelax=0.1
            NumTimeSteps=30
      dtime=60.0
            dtime=dtime/(hei*hei/KinVisc)
      times=0.05
            timef=0.0
      tini=0.5
```

```
c        Form the grid
     Call Grid_form(N,M,P,len,bre,hei)
c    set Initial conditon for  pressure, velocity, temperature, concentration
*      open (unit=3,file='in0.dat',status='UNKNOWN')
     open (unit=4,file='oscil.dat',status='NEW')
*     read(3,*) u,v,w,Temp,Press
********************************
* Initial Conditions
********************************
         DO i=0,N
          DO j=0,M+1
           DO k=0,P+1
                 u(i,j,k)=0.0
         ENDDO
         ENDDO
         ENDDO

         DO i=0,N+1
          DO j=0,M
           DO k=0,P+1
                 v(i,j,k)=0.0
         ENDDO
         ENDDO
         ENDDO

         DO i=0,N+1
          DO j=0,M+1
           DO k=0,P
                 w(i,j,k)=0.0
         ENDDO
         ENDDO
         ENDDO

         DO i=0,N+1
          DO j=0,M+1
           DO k=0,P+1
                 temp(i,j,k)=1.0
         ENDDO
         ENDDO
         ENDDO

         DO i=0,N+1
          DO j=0,M+1
*                temp(i,j,0)=0.5
                 temp(i,j,P+1)=0.0
         ENDDO
         ENDDO
         TimeStep=1
         do 15000 while (TimeStep.lt.NumTimeSteps)
         timef=Real(TimeStep)*dtime  !timef+dtime
   write(wc,*)"time=",timef,"RealTime=",timef*(hei*hei/KinVisc),"sec"
   do 301 i=0,N,1
      do 301 j=0,M+1,1
         do 301 k=0,P+1,1
           uo(i,j,k)=u(i,j,k)
           uinitial(i,j,k)=0.
```

```
 301 continue

     do 501 i=0,N+1,1
        do 501 j=0,M,1
           do 501 k=0,P+1,1
              vo(i,j,k)=v(i,j,k)
              vinitial(i,j,k)=0.

 501 continue

     do 454 k=0,P,1
        do 454 j=0,M+1,1
           do 454 i=0,N+1,1
              wo(i,j,k)=w(i,j,k)
              winitial(i,j,k)=0.
 454 continue


     do 748 k=0,P+1,1
        do 748 i=0,N+1,1
           do 748 j=0,M+1,1
              tempo(i,j,k)=temp(i,j,k)
 748 continue

     cut=1.0

     con2=0
     do 10000 while (con2.lt.100.and.(cut.gt.1E-5.or.cvt.gt.1E-5.
    &                 or.cwt.gt.1E-5))
     con2=con2+1
     converge=1.0
     con1=0
     do 3300 while (converge.GT.1.0e-8.and.con1.lt.100)
     uvwtime=0
     uvwtmax=20
     uvwerror=1.0
c    Repeat unitl the velocities have converged to a specific limit
     do 3000 while (uvwerror.gt.1.0e-8.and.uvwtime.lt.uvwtmax)
C*****************************************************************************************
*
c******************** Solving for velocity in the x direction ************************
C*****************************************************************************************
*
     uvwtime=uvwtime + 1
      do 1800 i=0,N,1
         do 1800 j=0,M+1,1
          do 1800 k=0,P+1,1
             uold(i,j,k)=u(i,j,k)

 1800 continue
c    Define the matrices for carrying out the thomas algorithm
c***************************************************************************************
c    For the lowest plane
c    For the first row of elements
c    For the lowest plane
     do 1005 j=0,N-2,1
```

```
c     For the first row of elements    15151515
       dZp=Zcord(1)-Zcord(0)
       dYp=ycord(1)-ycord(0)
       dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
       dXe=xcord(j+2)-xcord(j+1)
       dXw=xcord(j+1)-xcord(j)
       dYs=(ycord(1)-ycord(0))/2
       dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
       dZt=(zcord(2)+zcord(1))/2-(zcord(1)+zcord(0))/2
       dZb=(zcord(1)-zcord(0))/2
       De=dm1*dYp*dZp/dXe
       Dw=dm1*dYp*dZp/dXw
       Dn=dm1*dXp*dZp/dYn
       Ds=dm1*dXp*dZp/dYs
       Dt=dm1*dXp*dYp/dZt
       Db=dm1*dXp*dYp/dZb
       Fe=dYp*dZp*(u(j+1,1,1)+u(j+2,1,1))/2
       Fw=dYp*dZp*(u(j+1,1,1)+u(j,1,1))/2
       Fn=dXp*dZp*velonew(v(j+1,1,1),dxw/2,v(j+2,1,1),dxe/2)
       Fs=dXp*dZp*velonew(v(j+1,0,1),dxw/2,v(j+2,0,1),dxe/2)
       Fb=dXp*dYp*velonew(w(j+1,1,0),dxw/2,w(j+2,1,0),dxe/2)
       Ft=dXp*dYp*velonew(w(j+1,1,1),dxw/2,w(j+2,1,1),dxe/2)
       Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
       Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
       An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
       As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
       At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
       Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
       Apo=dXp*dYp*dZp/dtime
           f=0
           tempc=velonew(temp(1,1,f+1),dZb/2,temp(1,1,f+2),dZt/2)
       Ap=Ae+Aw+An+As+Ab+At+Apo
       Apu(j,0,0)=Ap
       b(0)=Ap
       c(0)=-An
c       u(j+1,1,0)=u(j+1,1,1)
c       u(j+1,0,1)=u(j+1,1,1)
      d(0)=Aw*u(j,1,1)+As*u(j+1,0,1)+Ae*u(j+2,1,1)+At*u(j+1,1,2)+
     & Ab*u(j+1,1,0)+dYp*dZp*(Press(j,0,0)-Press(j+1,0,0))
     & + Apo*uo(j+1,1,1)+Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)

c     for middle elements in the first row
      do 800 k=2,M-1,1
       dYp=ycord(k)-ycord(k-1)
       dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
       dZp=Zcord(1)-Zcord(0)
       dXe=xcord(j+2)-xcord(j+1)
       dXw=xcord(j+1)-xcord(j)
       dYs=(ycord(k)+ycord(k-1))/2 -(ycord(k-1)+ycord(k-2))/2
       dYn=(ycord(k+1)+ycord(k))/2-(ycord(k)+ycord(k-1))/2
       dZt=(zcord(2)+zcord(1))/2-(zcord(1)+zcord(0))/2
       dZb=(zcord(1)-zcord(0))/2
       De=dm1*dYp*dZp/dXe
       Dw=dm1*dYp*dZp/dXw
       Dn=dm1*dXp*dZp/dYn
       Ds=dm1*dXp*dZp/dYs
```

```
          Dt=dm1*dXp*dYp/dZt
          Db=dm1*dXp*dYp/dZb
          Fe=dYp*dZp*(u(j+1,k,1)+u(j+2,k,1))/2
          Fw=dYp*dZp*(u(j+1,k,1)+u(j,k,1))/2
          Fn=dXp*dZp*velonew(v(j+2,k,1),dxe/2,v(j+1,k,1),dxw/2)
          Fs=dXp*dZp*velonew(v(j+2,k-1,1),dxe/2,v(j+1,k-1,1),dxw/2)
          Fb=dXp*dYp*velonew(w(j+1,k,0),dxw/2,w(j+2,k,0),dxe/2)
          Ft=dXp*dYp*velonew(w(j+1,k,1),dxw/2,w(j+2,k,1),dxe/2)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
             tempc=velonew(temp(k,1,f+1),dZb/2,temp(k,1,f+2),dZt/2)
          Ap=Ae+Aw+An+As+At+Ab+Apo
          Apu(j,k-1,0)=Ap
          a(k-1)=-As
          b(k-1)=Ap
          c(k-1)=-An
c          u(j+1,k,0)=u(j+1,k,1)
          d(k-1)=Aw*u(j,k,1)+Ae*u(j+2,k,1)+At*u(j+1,k,2)+Ab*u(j+1,k,0)
     &     +dYp*dZp*(Press(j,k-1,0)-Press(j+1,k-1,0)) + Apo*uo(j+1,k,1)
     &                 +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)

 800  continue
c     For last element in the first row
        dYp=ycord(M)-ycord(M-1)
        dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
        dZp=Zcord(1)-Zcord(0)
        dXe=xcord(j+2)-xcord(j+1)
        dXw=xcord(j+1)-xcord(j)
        dYs=(ycord(M)+ycord(M-1))/2-(ycord(M-1)+ycord(M-2))/2
        dYn=(ycord(M)-ycord(M-1))/2
        dZt=(zcord(2)+zcord(1))/2-(zcord(1)+zcord(0))/2
        dZb=(zcord(1)-zcord(0))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*(u(j+1,M,1)+u(j+2,M,1))/2
        Fw=dYp*dZp*(u(j+1,M,1)+u(j,M,1))/2
        Fn=dXp*dZp*velonew(v(j+2,M,1),dxe/2,v(j+1,M,1),dxw/2)
        Fs=dXp*dZp*velonew(v(j+2,M-1,1),dxe/2,v(j+1,M-1,1),dxw/2)
        Fb=dXp*dYp*velonew(w(j+1,M,0),dxw/2,w(j+2,M,0),dxe/2)
        Ft=dXp*dYp*velonew(w(j+1,M,1),dxw/2,w(j+2,M,1),dxe/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
             tempc=velonew(temp(N,1,f+1),dZb/2,temp(N,1,f+2),dZt/2)
        Apu(j,M-1,0)=Ap
        a(M-1)=-As
        b(M-1)=Ap
c        u(j+1,M+1,1)=u(j+1,M,1)
c        u(j+1,M,0)=u(j+1,M,1)
        d(M-1)=Ae*u(j+2,M,1)+Aw*u(j,M,1)+An*u(j+1,M+1,1)+
     &    At*u(j+1,M,2)+Ab*u(j+1,M,0) +
     &   dYp*dZp*(Press(j,M-1,0)-Press(j+1,M-1,0))+Apo*uo(j+1,M,1)
     &   +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)

c     Call the Thomas Algorithm subprogram
        Call Thomas_algo(M)
        do 900 i=1,M,1
            u(j+1,i,1)=(1.-urelax)*u(j+1,i,1) + urelax*x(i)
c            write(wc,*),j+1,i,1,u(j+1,i,1)
 900   continue
 1005  continue

c     For middle plane
      do 1010 f=1,P-2,1
      do 1008 j=0,N-2,1
c     For the first row of elements
       dZp=Zcord(f+1)-Zcord(f)
       dYp=ycord(1)-ycord(0)
       dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
       dXe=xcord(j+2)-xcord(j+1)
       dXw=xcord(j+1)-xcord(j)
       dYs=(ycord(1)-ycord(0))/2
       dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
       dZt=(zcord(f+2)+zcord(f+1))/2-(zcord(f+1)+zcord(f))/2
       dZb=(zcord(f+1)+zcord(f))/2-(zcord(f)+zcord(f-1))/2
       De=dm1*dYp*dZp/dXe
       Dw=dm1*dYp*dZp/dXw
       Dn=dm1*dXp*dZp/dYn
       Ds=dm1*dXp*dZp/dYs
       Dt=dm1*dXp*dYp/dZt
       Db=dm1*dXp*dYp/dZb
       Fe=dYp*dZp*(u(j+1,1,f+1)+u(j+2,1,f+1))/2
       Fw=dYp*dZp*(u(j+1,1,f+1)+u(j,1,f+1))/2
       Fn=dXp*dZp*velonew(v(j+1,1,f+1),dxw/2,v(j+2,1,f+1),dxe/2)
       Fs=dXp*dZp*velonew(v(j+1,0,f+1),dxw/2,v(j+2,0,f+1),dxe/2)
       Fb=dXp*dYp*velonew(w(j+1,1,f),dxw/2,w(j+2,1,f),dxe/2)
       Ft=dXp*dYp*velonew(w(j+1,1,f+1),dxw/2,w(j+2,1,f+1),dxe/2)
       Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
       Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
       An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
       As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
       At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
       Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
       Apo=dXp*dYp*dZp/dtime
            tempc=velonew(temp(1,j+1,f+1),dZb/2,temp(1,j+1,f+2),dZt/2)
       Ap=Ae+Aw+An+As+Ab+At+Apo
       Apu(j,0,f)=Ap
       b(0)=Ap
```

```
        c(0)=-An

c        u(j+1,0,f+1)=u(j+1,1,f+1)
        d(0)=Aw*u(j,1,f+1)+As*u(j+1,0,f+1)+Ae*u(j+2,1,f+1)+
     &  At*u(j+1,1,f+2)+ Ab*u(j+1,1,f)+
     &  dYp*dZp*(Press(j,0,f)-Press(j+1,0,f))+ Apo*uo(j+1,1,f+1)
     &  +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)

c     for middle elements in the first row
        do 810 k=2,M-1,1
          dYp=ycord(k)-ycord(k-1)
          dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
          dZp=Zcord(f+1)-Zcord(f)
          dXe=xcord(j+2)-xcord(j+1)
          dXw=xcord(j+1)-xcord(j)
          dYs=(ycord(k)+ycord(k-1))/2 -(ycord(k-1)+ycord(k-2))/2
          dYn=(ycord(k+1)+ycord(k))/2-(ycord(k)+ycord(k-1))/2
          dZt=(zcord(f+2)+zcord(f+1))/2-(zcord(f+1)+zcord(f))/2
          dZb=(zcord(f+1)+zcord(f))/2-(zcord(f)+zcord(f-1))/2
          De=dm1*dYp*dZp/dXe
          Dw=dm1*dYp*dZp/dXw
          Dn=dm1*dXp*dZp/dYn
          Ds=dm1*dXp*dZp/dYs
          Dt=dm1*dXp*dYp/dZt
          Db=dm1*dXp*dYp/dZb
          Fe=dYp*dZp*(u(j+1,k,f+1)+u(j+2,k,f+1))/2
          Fw=dYp*dZp*(u(j+1,k,f+1)+u(j,k,f+1))/2
          Fn=dXp*dZp*velonew(v(j+2,k,f+1),dxe/2,v(j+1,k,f+1),dxw/2)
          Fs=dXp*dZp*velonew(v(j+2,k-1,f+1),
     &     dxe/2,v(j+1,k-1,f+1),dxw/2)
          Fb=dXp*dYp*velonew(w(j+1,k,f),dxw/2,w(j+2,k,f),dxe/2)
          Ft=dXp*dYp*velonew(w(j+1,k,f+1),dxw/2,w(j+2,k,f+1),dxe/2)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
          Apu(j,k-1,f)=Ap
          a(k-1)=-As
          b(k-1)=Ap
          c(k-1)=-An
       tempc=velonew(temp(k,j+1,f+1),dZb/2,temp(k,j+1,f+2),dZt/2)
          d(k-1)=Aw*u(j,k,f+1)+Ae*u(j+2,k,f+1)+At*u(j+1,k,f+2)
     &     + Ab*u(j+1,k,f) + dYp*dZp*(Press(j,k-1,f)-Press(j+1,k-1,f))
     &     + Apo*uo(j+1,k,f+1)+Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)


 810  continue
c    For last element in the first row
       dYp=ycord(M)-ycord(M-1)
       dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
       dZp=Zcord(f+1)-Zcord(f)
       dXe=xcord(j+2)-xcord(j+1)
```

```
         dXw=xcord(j+1)-xcord(j)
         dYs=(ycord(M)+ycord(M-1))/2-(ycord(M-1)+ycord(M-2))/2
         dYn=(ycord(M)-ycord(M-1))/2
         dZt=(zcord(f+2)+zcord(f+1))/2-(zcord(f+1)+zcord(f))/2
         dZb=(zcord(f+1)+zcord(f))/2-(zcord(f)+zcord(f-1))/2
         De=dm1*dYp*dZp/dXe
         Dw=dm1*dYp*dZp/dXw
         Dn=dm1*dXp*dZp/dYn
         Ds=dm1*dXp*dZp/dYs
         Dt=dm1*dXp*dYp/dZt
         Db=dm1*dXp*dYp/dZb
         Fe=dYp*dZp*(u(j+1,M,f+1)+u(j+2,M,f+1))/2
         Fw=dYp*dZp*(u(j+1,M,f+1)+u(j,M,f+1))/2
         Fn=dXp*dZp*velonew(v(j+2,M,f+1),dxe/2,v(j+1,M,f+1),dxw/2)
         Fs=dXp*dZp*velonew(v(j+2,M-1,f+1),
     &    dxe/2,v(j+1,M-1,f+1),dxw/2)
         Fb=dXp*dYp*velonew(w(j+1,M,f),dxw/2,w(j+2,M,f),dxe/2)
         Ft=dXp*dYp*velonew(w(j+1,M,f+1),dxw/2,w(j+2,M,f+1),dxe/2)
         Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
         Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
         An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
         As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
         At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
         Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
         Apo=dXp*dYp*dZp/dtime
         Ap=Ae+Aw+An+As+At+Ab+Apo
         Apu(j,M-1,f)=Ap
         a(M-1)=-As
         b(M-1)=Ap
c         u(j+1,M+1,f+1)=u(j+1,M,f+1)
        tempc=velonew(temp(N,j+1,f+1),dZb/2,temp(N,j+1,f+2),dZt/2)
         d(M-1)=Ae*u(j+2,M,f+1)+Aw*u(j,M,f+1)+An*u(j+1,M+1,f+1)+
     &     At*u(j+1,M,f+2)+Ab*u(j+1,M,f) +
     &     dYp*dZp*(Press(j,M-1,f)-Press(j+1,M-1,f))+Apo*uo(j+1,M,f+1)
     &     +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)


c     Call the Thomas Algorithm subprogram
         Call Thomas_algo(M)
         do 910 i=1,M,1
               u(j+1,i,f+1)=(1.-urelax)*u(j+1,i,f+1) + urelax*x(i)
c                write(wc,*) j+1,i,f+1,u(j+1,i,f+1)
 910   continue
 1008  continue
 1010 continue
c      For the top most plane
        do 1018 j=0,N-2,1
c      For the first row of elements
         dZp=Zcord(P)-Zcord(P-1)
         dYp=ycord(1)-ycord(0)
         dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
         dXe=xcord(j+2)-xcord(j+1)
         dXw=xcord(j+1)-xcord(j)
         dYs=(ycord(1)-ycord(0))/2
         dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
         dZt=(zcord(P)-zcord(P-1))/2
        dZb=(zcord(P)+zcord(P-1))/2-(zcord(P-1)+zcord(P-2))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*(u(j+1,1,P)+u(j+2,1,P))/2
        Fw=dYp*dZp*(u(j+1,1,P)+u(j,1,P))/2
        Fn=dXp*dZp*velonew(v(j+1,1,P),dxw/2,v(j+2,1,P),dxe/2)
        Fs=dXp*dZp*velonew(v(j+1,0,P),dxw/2,v(j+2,0,P),dxe/2)
        Fb=dXp*dYp*velonew(w(j+1,1,P-1),dxw/2,w(j+2,1,P-1),dxe/2)
        Ft=dXp*dYp*velonew(w(j+1,1,P),dxw/2,w(j+2,1,P),dxe/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+Ab+At+Apo
        Apu(j,0,P-1)=Ap
        b(0)=Ap
        c(0)=-An
        u(j+1,1,P+1)=u(j+1,1,P)
c        u(j+1,0,P)=u(j+1,1,P)
            tempc=velonew(temp(1,M,f+1),dZb/2,temp(1,M,f+2),dZt/2)
        d(0)=Aw*u(j,1,P)+As*u(j+1,0,P)+Ae*u(j+2,1,P)+
     &  At*u(j+1,1,P+1)+ Ab*u(j+1,1,P-1)+
     &  dYp*dZp*(Press(j,0,P-1)-Press(j+1,0,P-1))+ Apo*uo(j+1,1,P)
     &   +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)

c     for middle elements in the first row
        do 820 k=2,M-1,1
         dYp=ycord(k)-ycord(k-1)
         dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
         dZp=Zcord(P)-Zcord(P-1)
         dXe=xcord(j+2)-xcord(j+1)
         dXw=xcord(j+1)-xcord(j)
         dYs=(ycord(k)+ycord(k-1))/2 -(ycord(k-1)+ycord(k-2))/2
         dYn=(ycord(k+1)+ycord(k))/2-(ycord(k)+ycord(k-1))/2
         dZt=(zcord(P)-zcord(P-1))/2
         dZb=(zcord(P)+zcord(P-1))/2-(zcord(P-1)+zcord(P-2))/2
         De=dm1*dYp*dZp/dXe
         Dw=dm1*dYp*dZp/dXw
         Dn=dm1*dXp*dZp/dYn
         Ds=dm1*dXp*dZp/dYs
         Dt=dm1*dXp*dYp/dZt
         Db=dm1*dXp*dYp/dZb
         Fe=dYp*dZp*(u(j+1,k,P)+u(j+2,k,P))/2
         Fw=dYp*dZp*(u(j+1,k,P)+u(j,k,P))/2
         Fn=dXp*dZp*velonew(v(j+2,k,P),dxe/2,v(j+1,k,P),dxw/2)
         Fs=dXp*dZp*velonew(v(j+2,k-1,P),
     &    dxe/2,v(j+1,k-1,P),dxw/2)
         Fb=dXp*dYp*velonew(w(j+1,k,P-1),dxw/2,w(j+2,k,P-1),dxe/2)
         Ft=dXp*dYp*velonew(w(j+1,k,P),dxw/2,w(j+2,k,P),dxe/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        Apu(j,k-1,P-1)=Ap
        a(k-1)=-As
        b(k-1)=Ap
        c(k-1)=-An
        u(j+1,k,P+1)=u(j+1,k,P)
          tempc=velonew(temp(k,M,f+1),dZb/2,temp(k,M,f+2),dZt/2)
        d(k-1)=Aw*u(j,k,P)+Ae*u(j+2,k,P)+At*u(j+1,k,P+1)
     &     + Ab*u(j+1,k,P-1) +
     &      dYp*dZp*(Press(j,k-1,P-1)-Press(j+1,k-1,P-1))
     &     + Apo*uo(j+1,k,P)+Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)


  820  continue
c    For last element in the first row
       dYp=ycord(M)-ycord(M-1)
       dXp=(xcord(j+2)+xcord(j+1))/2-(xcord(j+1)+xcord(j))/2
       dZp=Zcord(P)-Zcord(P-1)
       dXe=xcord(j+2)-xcord(j+1)
       dXw=xcord(j+1)-xcord(j)
       dYs=(ycord(M)+ycord(M-1))/2-(ycord(M-1)+ycord(M-2))/2
       dYn=(ycord(M)-ycord(M-1))/2
       dZt=(zcord(P)-zcord(P-1))/2
       dZb=(zcord(P)+zcord(P-1))/2-(zcord(P-1)+zcord(P-2))/2
       De=dm1*dYp*dZp/dXe
       Dw=dm1*dYp*dZp/dXw
       Dn=dm1*dXp*dZp/dYn
       Ds=dm1*dXp*dZp/dYs
       Dt=dm1*dXp*dYp/dZt
       Db=dm1*dXp*dYp/dZb
       Fe=dYp*dZp*(u(j+1,M,P)+u(j+2,M,P))/2
       Fw=dYp*dZp*(u(j+1,M,P)+u(j,M,P))/2
       Fn=dXp*dZp*velonew(v(j+2,M,P),dxe/2,v(j+1,M,P),dxw/2)
       Fs=dXp*dZp*velonew(v(j+2,M-1,P),
     &    dxe/2,v(j+1,M-1,P),dxw/2)
       Fb=dXp*dYp*velonew(w(j+1,M,P-1),dxw/2,w(j+2,M,P-1),dxe/2)
       Ft=dXp*dYp*velonew(w(j+1,M,P),dxw/2,w(j+2,M,P),dxe/2)
       Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
       Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
       An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
       As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
       At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
       Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
       Apo=dXp*dYp*dZp/dtime
       Ap=Ae+Aw+An+As+At+Ab+Apo
       Apu(j,M-1,P-1)=Ap
       a(M-1)=-As
       b(M-1)=Ap
c       u(j+1,M+1,P)=u(j+1,M,P)
```

```
      u(j+1,M,P+1)=u(j+1,M,P)
           tempc=velonew(temp(N,M,f+1),dZb/2,temp(N,M,f+2),dZt/2)
     d(M-1)=Ae*u(j+2,M,P)+Aw*u(j,M,P)+An*u(j+1,M+1,P)+
   &     At*u(j+1,M,P+1)+Ab*u(j+1,M,P-1) +
   &    dYp*dZp*(Press(j,M-1,P-1)-Press(j+1,M-1,P-1))+Apo*uo(j+1,M,P)
   &    +Re*dm2*(tempc-tini)*dXp*dYp*dZp*sin(Tilt)


c    Call the Thomas Algorithm subprogram
      Call Thomas_algo(M)
      do 920 i=1,M,1
           u(j+1,i,P)=(1.-urelax)*u(j+1,i,P) + urelax*x(i)
c             write(wc,*), j+1,i,P,u(j+1,i,P)
 920  continue
 1018  continue


C***********************************************************************
C*************END OF CALCULATIONS FOR U VELOCITY********************



      uerrorn=0.
      uerrord=0.
      do 1500 i=0,N,1
         do 1600 j=0,M+1,1
           DO 1651 k=0,P+1,1
              uerrorn=uerrorn+abs(u(i,j,k)-uold(i,j,k))
              uerrord=uerrord+abs(u(i,j,k))
 1651         continue
 1600       continue
 1500   continue
      if (uerrord.lt.0.0000001) then
              uerror=-uerrord
           else
              uerror=uerrorn/uerrord
           end if
c      write(wc,*)"uerror = " ,uerror

c******************************************************************************
c*******************Solving for velocity in the y direction**************
c******************************************************************************
      do 2600 i=0,N+1,1
        do 2600 j=0,M,1
           do 2600 k=0,P+1,1
              vold(i,j,k)=v(i,j,k)

 2600  continue
c    For the lowest plane of the liquid
c    for the  rows
c    for the first element
      do 2000 j=1,M-1,1
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(1)-xcord(0)
        dZp=Zcord(1)-Zcord(0)
        dXe=(xcord(2)+xcord(1))/2-(xcord(1)+xcord(0))/2
```

```
         dXw=(xcord(1)-xcord(0))/2
         dYs=ycord(j)-ycord(j-1)
         dYn=ycord(j+1)-ycord(j)
         dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
         dZb=(Zcord(1)-Zcord(0))/2
         De=dm1*dYp*dZp/dXe
         Dw=dm1*dYp*dZp/dXw
         Dn=dm1*dXp*dZp/dYn
         Ds=dm1*dXp*dZp/dYs
         Dt=dm1*dXp*dYp/dZt
         Db=dm1*dXp*dYp/dZb
         Fe=dYp*dZp*velonew(u(1,j,1),dys/2,u(1,j+1,1),dyn/2)
         Fw=dYp*dZp*velonew(u(0,j,1),dys/2,u(0,j+1,1),dyn/2)
         Fn=dXp*dZp*(v(1,j,1)+v(1,j+1,1))/2
         Fs=dXp*dZp*(v(1,j,1)+v(1,j-1,1))/2
         Ft=dXp*dYp*velonew(w(1,j,1),dYs/2,w(1,j+1,1),dYn/2)
         Fb=dXp*dYp*velonew(w(1,j,0),dYs/2,w(1,j+1,0),dYn/2)
         Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
         Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
         An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
         As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
         At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
         Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
         Apo=dXp*dYp*dZp/dtime
         Ap=Ae+Aw+An+As+At+Ab+Apo
         Apv(0,j-1,0)=Ap
         b(0)=Ap
         c(0)=-Ae
c         v(0,j,1)=v(1,j,1)
c         v(1,j,0)=v(1,j,1)
         d(0)=Aw*v(0,j,1)+As*v(1,j-1,1)+An*v(1,j+1,1)+
   &          At*v(1,j,2)+ Ab*v(1,j,0) +
   &       dXp*dZp*(Press(0,j-1,0)-Press(0,j,0))+ Apo*vo(1,j,1)


c    For the middle elements
         do 2800 k=2,N-1,1
            dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
            dXp=xcord(k)-xcord(k-1)
            dZp=Zcord(1)-Zcord(0)
            dXe=(xcord(k+1)+xcord(k))/2-(xcord(k)+xcord(k-1))/2
            dXw=(xcord(k)+xcord(k-1))/2-(xcord(k-1)+xcord(k-2))/2
            dYs=ycord(j)-ycord(j-1)
            dYn=ycord(j+1)-ycord(j)
            dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
            dZb=(Zcord(1)-Zcord(0))/2
            De=dm1*dYp*dZp/dXe
            Dw=dm1*dYp*dZp/dXw
            Dn=dm1*dXp*dZp/dYn
            Ds=dm1*dXp*dZp/dYs
            Dt=dm1*dXp*dYp/dZt
            Db=dm1*dXp*dYp/dZb
            Fe=dYp*dZp*velonew(u(k,j,1),
   &           dys/2,u(k,j+1,1),dyn/2)
            Fw=dYp*dZp*velonew(u(k-1,j,1),dys/2,
   &           u(k-1,j+1,1),dyn/2)
```

```
            Fn=dXp*dZp*(v(k,j,1)+v(k,j+1,1))/2
            Fs=dXp*dZp*(v(k,j,1)+v(k,j-1,1))/2
            Ft=dXp*dYp*velonew(w(k,j,1),dYs/2,
     &        w(k,j+1,1),dYn/2)
            Fb=dXp*dYp*velonew(w(k,j,0),dYs/2,
     &        w(k,j+1,0),dYn/2)
            Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
            Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
            An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
            As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
            At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
            Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
            Apo=dXp*dYp*dZp/dtime
            Ap=Ae+Aw+An+As+At+Ab+Apo
            Apv(k-1,j-1,0)=Ap
            a(k-1)=-Aw
            b(k-1)=Ap
            c(k-1)=-Ae
c            v(k,j,0)=v(k,j,1)
            d(k-1)=As*v(k,j-1,1)+An*v(k,j+1,1)+
     &          At*v(k,j,2)+Ab*v(k,j,0)+
     &    dXp*dZp*(Press(k-1,j-1,0)-Press(k-1,j,0)) +Apo*vo(k,j,1)


2800    continue
c    For the last element in the row
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(N)-xcord(N-1)
        dZp=Zcord(1)-Zcord(0)
        dXe=(xcord(N)-xcord(N-1))/2
        dXw=(xcord(N)+xcord(N-1))/2-(xcord(N-1)+xcord(N-2))/2
        dYs=ycord(j)-ycord(j-1)
        dYn=ycord(j+1)-ycord(j)
        dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
        dZb=(Zcord(1)-Zcord(0))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(N,j,1),dys/2,u(N,j+1,1),dyn/2)
        Fw=dYp*dZp*velonew(u(N-1,j,1),dys/2,u(N-1,j+1,1),dyn/2)
        Fn=dXp*dZp*(v(N,j,1)+v(N,j+1,1))/2
        Fs=dXp*dZp*(v(N,j,1)+v(N,j-1,1))/2
        Ft=dXp*dYp*velonew(w(N,j,1),dYs/2,
     &      w(N,j+1,1),dYn/2)
        Fb=dXp*dYp*velonew(w(N,j,0),dYs/2,
     &      w(N,j+1,0),dYn/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
```

```
         Apo=dXp*dYp*dZp/dtime
         Ap=Ae+Aw+An+As+At+Ab+Apo
         Apv(N-1,j-1,0)=Ap
         a(N-1)=-Aw
         b(N-1)=Ap

c         v(N+1,j,1)=v(N,j,1)
c         v(N,j,0)=v(N,j,1)
        d(N-1)=Ae*v(N+1,j,1)+As*v(N,j-1,1)+An*v(N,j+1,1)+
     &      At*v(N,j,2) + Ab*v(N,j,0) +
     &    dXp*dZp*(Press(N-1,j-1,0)-Press(N-1,j,0))+ Apo*vo(N,j,1)


c    Call the Thomas Algorithm subprogram
        Call Thomas_algo(N)
        do 2900 k=1,N,1
          v(k,j,1)=(1.-vrelax)*v(k,j,1) + vrelax*x(k)
 2900    continue
 2000   continue
c***********************************************************
c    for the middle planes of the liquid
     do 2044 f=1,P-2,1
c    for the  rows
c    for the first element
      do 2002 j=1,M-1,1
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(1)-xcord(0)
        dZp=Zcord(f+1)-Zcord(f)
        dXe=(xcord(2)+xcord(1))/2-(xcord(1)+xcord(0))/2
        dXw=(xcord(1)-xcord(0))/2
        dYs=ycord(j)-ycord(j-1)
        dYn=ycord(j+1)-ycord(j)
        dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(1,j,f+1),dys/2,u(1,j+1,f+1),dyn/2)
        Fw=dYp*dZp*velonew(u(0,j,f+1),dys/2,u(0,j+1,f+1),dyn/2)
        Fn=dXp*dZp*(v(1,j,f+1)+v(1,j+1,f+1))/2
        Fs=dXp*dZp*(v(1,j,f+1)+v(1,j-1,f+1))/2
        Ft=dXp*dYp*velonew(w(1,j,f+1),dYs/2,w(1,j+1,f+1),dYn/2)
        Fb=dXp*dYp*velonew(w(1,j,f),dYs/2,w(1,j+1,f),dYn/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        Apv(0,j-1,f)=Ap
        b(0)=Ap
```

```
         c(0)=-Ae
c          v(0,j,f+1)=v(1,j,f+1)
         d(0)=Aw*v(0,j,f+1)+As*v(1,j-1,f+1)+An*v(1,j+1,f+1)+
     &          At*v(1,j,f+2)+ Ab*v(1,j,f) +
     &   dXp*dZp*(Press(0,j-1,f)-Press(0,j,f))+ Apo*vo(1,j,f+1)


c     For the middle elements
         do 2802 k=2,N-1,1
            dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
            dXp=xcord(k)-xcord(k-1)
            dZp=Zcord(f+1)-Zcord(f)
            dXe=(xcord(k+1)+xcord(k))/2-(xcord(k)+xcord(k-1))/2
            dXw=(xcord(k)+xcord(k-1))/2-(xcord(k-1)+xcord(k-2))/2
            dYs=ycord(j)-ycord(j-1)
            dYn=ycord(j+1)-ycord(j)
            dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
            dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
            De=dm1*dYp*dZp/dXe
            Dw=dm1*dYp*dZp/dXw
            Dn=dm1*dXp*dZp/dYn
            Ds=dm1*dXp*dZp/dYs
            Dt=dm1*dXp*dYp/dZt
            Db=dm1*dXp*dYp/dZb
            Fe=dYp*dZp*velonew(u(k,j,f+1),dys/2,
     &        u(k,j+1,f+1),dyn/2)
            Fw=dYp*dZp*velonew(u(k-1,j,f+1),dys/2,
     &           u(k-1,j+1,f+1),dyn/2)
            Fn=dXp*dZp*(v(k,j,f+1)+v(k,j+1,f+1))/2
            Fs=dXp*dZp*(v(k,j,f+1)+v(k,j-1,f+1))/2
            Ft=dXp*dYp*velonew(w(k,j,f+1),dYs/2,
     &        w(k,j+1,f+1),dYn/2)
            Fb=dXp*dYp*velonew(w(k,j,f),dYs/2,
     &        w(k,j+1,f),dYn/2)
            Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
            Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
            An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
            As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
            At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
            Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
            Apo=dXp*dYp*dZp/dtime
            Ap=Ae+Aw+An+As+At+Ab+Apo
            Apv(k-1,j-1,f)=Ap
            a(k-1)=-Aw
            b(k-1)=Ap
            c(k-1)=-Ae
            d(k-1)=As*v(k,j-1,f+1)+An*v(k,j+1,f+1)+
     &             At*v(k,j,f+2)+Ab*v(k,j,f)+
     &    dXp*dZp*(Press(k-1,j-1,f)-Press(k-1,j,f)) + Apo*vo(k,j,f+1)


 2802    continue
c     For the last element in the row
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(N)-xcord(N-1)
        dZp=Zcord(f+1)-Zcord(f)
        dXe=(xcord(N)-xcord(N-1))/2
        dXw=(xcord(N)+xcord(N-1))/2-(xcord(N-1)+xcord(N-2))/2
        dYs=ycord(j)-ycord(j-1)
        dYn=ycord(j+1)-ycord(j)
        dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dZb=(Zcord(f+1)+Zcord(f))-(Zcord(f)+Zcord(f-1))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(N,j,f+1),dys/2,u(N,j+1,f+1),dyn/2)
        Fw=dYp*dZp*velonew(u(N-1,j,f+1),
     &          dys/2,u(N-1,j+1,f+1),dyn/2)
        Fn=dXp*dZp*(v(N,j,f+1)+v(N,j+1,f+1))/2
        Fs=dXp*dZp*(v(N,j,f+1)+v(N,j-1,f+1))/2
        Ft=dXp*dYp*velonew(w(N,j,f+1),dYs/2,
     &          w(N,j+1,f+1),dYn/2)
        Fb=dXp*dYp*velonew(w(N,j,f),dYs/2,
     &          w(N,j+1,f),dYn/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        Apv(N-1,j-1,f)=Ap
        a(N-1)=-Aw
        b(N-1)=Ap
c        v(N+1,j,f+1)=v(N,j,f+1)
        d(N-1)=Ae*v(N+1,j,f+1)+As*v(N,j-1,f+1)+An*v(N,j+1,f+1)+
     &      At*v(N,j,f+2) + Ab*v(N,j,f) +
     &    dXp*dZp*(Press(N-1,j-1,f)-Press(N-1,j,f))+ Apo*vo(N,j,f+1)


c    Call the Thomas Algorithm subprogram
       Call Thomas_algo(N)
       do 2902 k=1,N,1
          v(k,j,f+1)=(1.-vrelax)*v(k,j,f+1) + vrelax*x(k)
 2902    continue
 2002   continue
c
 2044 continue
c***********************************************************
c    for the top most plane of the liquid
c    for the  rows
c    for the first element
      do 2005 j=1,M-1,1
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(1)-xcord(0)
        dZp=Zcord(P)-Zcord(P-1)
        dXe=(xcord(2)+xcord(1))/2-(xcord(1)+xcord(0))/2
        dXw=(xcord(1)-xcord(0))/2
        dYs=ycord(j)-ycord(j-1)
        dYn=ycord(j+1)-ycord(j)
        dZt=(Zcord(P)-Zcord(P-1))/2
        dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(1,j,P),dys/2,u(1,j+1,P),dyn/2)
        Fw=dYp*dZp*velonew(u(0,j,P),dys/2,u(0,j+1,P),dyn/2)
        Fn=dXp*dZp*(v(1,j,P)+v(1,j+1,P))/2
        Fs=dXp*dZp*(v(1,j,P)+v(1,j-1,P))/2
        Ft=dXp*dYp*velonew(w(1,j,P),dYs/2,w(1,j+1,P),dYn/2)
        Fb=dXp*dYp*velonew(w(1,j,P-1),dYs/2,w(1,j+1,P-1),dYn/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        Apv(0,j-1,P-1)=Ap
        b(0)=Ap
        c(0)=-Ae
c        v(0,j,P)=v(1,j,P)
        v(1,j,P+1)=v(1,j,P)
        d(0)=Aw*v(0,j,P)+As*v(1,j-1,P)+An*v(1,j+1,P)+
     &        At*v(1,j,P+1)+ Ab*v(1,j,P-1) +
     &  dXp*dZp*(Press(0,j-1,P-1)-Press(0,j,P-1)) +Apo*vo(1,j,P)


c     For the middle elements
        do 2805 k=2,N-1,1
            dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
            dXp=xcord(k)-xcord(k-1)
            dZp=Zcord(P)-Zcord(P-1)
            dXe=(xcord(k+1)+xcord(k))/2-(xcord(k)+xcord(k-1))/2
            dXw=(xcord(k)+xcord(k-1))/2-(xcord(k-1)+xcord(k-2))/2
            dYs=ycord(j)-ycord(j-1)
            dYn=ycord(j+1)-ycord(j)
            dZt=(Zcord(P)-Zcord(P-1))/2
            dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
            De=dm1*dYp*dZp/dXe
            Dw=dm1*dYp*dZp/dXw
            Dn=dm1*dXp*dZp/dYn
            Ds=dm1*dXp*dZp/dYs
            Dt=dm1*dXp*dYp/dZt
            Db=dm1*dXp*dYp/dZb
            Fe=dYp*dZp*velonew(u(k,j,P),dys/2,
     &        u(k,j+1,P),dyn/2)
```

```
          Fw=dYp*dZp*velonew(u(k-1,j,P),dys/2,
     &         u(k-1,j+1,P),dyn/2)
          Fn=dXp*dZp*(v(k,j,P)+v(k,j+1,P))/2
          Fs=dXp*dZp*(v(k,j,P)+v(k,j-1,P))/2
          Ft=dXp*dYp*velonew(w(k,j,P),dYs/2,
     &      w(k,j+1,P),dYn/2)
          Fb=dXp*dYp*velonew(w(k,j,P-1),dYs/2,
     &      w(k,j+1,P-1),dYn/2)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
          Apv(k-1,j-1,P-1)=Ap
          a(k-1)=-Aw
          b(k-1)=Ap
          c(k-1)=-Ae
          v(k,j,P+1)=v(k,j,P)
          d(k-1)=As*v(k,j-1,P)+An*v(k,j+1,P)+
     &         At*v(k,j,P+1)+Ab*v(k,j,P-1)+
     &    dXp*dZp*(Press(k-1,j-1,P-1)-Press(k-1,j,P-1))+ Apo*vo(k,j,P)


 2805   continue
c   For the last element in the row
        dYp=(ycord(j+1)+ycord(j))/2-(ycord(j)+ycord(j-1))/2
        dXp=xcord(N)-xcord(N-1)
        dZp=Zcord(P)-Zcord(P-1)
        dXe=(xcord(N)-xcord(N-1))/2
        dXw=(xcord(N)+xcord(N-1))/2-(xcord(N-1)+xcord(N-2))/2
        dYs=ycord(j)-ycord(j-1)
        dYn=ycord(j+1)-ycord(j)
        dZt=(Zcord(P)-Zcord(P-1))/2
        dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(N,j,P),dys/2,u(N,j+1,P),dyn/2)
        Fw=dYp*dZp*velonew(u(N-1,j,P),
     &       dys/2,u(N-1,j+1,P),dyn/2)
        Fn=dXp*dZp*(v(N,j,P)+v(N,j+1,P))/2
        Fs=dXp*dZp*(v(N,j,P)+v(N,j-1,P))/2
        Ft=dXp*dYp*velonew(w(N,j,P),dYs/2,
     &       w(N,j+1,P),dYn/2)
        Fb=dXp*dYp*velonew(w(N,j,P-1),dYs/2,
     &       w(N,j+1,P-1),dYn/2)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        Apv(N-1,j-1,P-1)=Ap
        a(N-1)=-Aw
        b(N-1)=Ap
c        v(N+1,j,P)=v(N,j,P)
        v(N,j,P+1)=v(N,j,P)
        d(N-1)=Ae*v(N+1,j,P)+As*v(N,j-1,P)+An*v(N,j+1,P)+
     &      At*v(N,j,P+1) + Ab*v(N,j,P-1) +
     &    dXp*dZp*(Press(N-1,j-1,P-1)-Press(N-1,j,P-1))+Apo*vo(N,j,P)


c    Call the Thomas Algorithm subprogram
       Call Thomas_algo(N)
       do 2905 k=1,N,1
         v(k,j,P)=(1.-vrelax)*v(k,j,P) + vrelax*x(k)
2905     continue
2005   continue

c***************End of v velocity****************************
      verrorn=0.
      verrord=0.
      do 2100 i=0,N+1,1
        do 2200 j=0,M,1
          DO 2251 K=0,P+1,1
            verrorn=verrorn+abs(v(i,j,k)-vold(i,j,k))
            verrord=verrord+abs(v(i,j,k))
2251        continue
2200      continue
2100    continue
      if (verrord.lt.0.0000001) then
           verror=-verrord
        else
          verror=verrorn/verrord
      end if
c     write(wc,*)"verror = " ,verror
c************************************************************
c*************solving for the w velocity*********************
c************************************************************


      do 1802 i=0,N+1,1
        do 1802 j=0,M+1,1
          do 1802 k=0,P,1
            wold(i,j,k)=w(i,j,k)

1802   continue
c    Define the matrices for carrying out the thomas algorithm
c****************************************************************************************
c    For all planes  16161616
c    For the first row of elements
c    For the first element
      do 3011 f=0,P-2,1
```

```
        dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dYp=ycord(1)-ycord(0)
        dXp=xcord(1)-xcord(0)
        dXe=(xcord(2)+xcord(1))/2-(Xcord(1)+Xcord(0))/2
        dXw=(xcord(1)-xcord(0))/2
        dYs=(ycord(1)-ycord(0))/2
        dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
        dZt=zcord(f+2)-zcord(f+1)
        dZb=zcord(f+1)-zcord(f)
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(1,1,f+1),dZb/2,u(1,1,f+2),dZt/2)
        Fw=dYp*dZp*velonew(u(0,1,f+1),dZb/2,u(0,1,f+2),dZt/2)
        Fn=dXp*dZp*velonew(v(1,1,f+1),dZb/2,v(1,1,f+2),dZt/2)
        Fs=dXp*dZp*velonew(v(1,0,f+1),dZb/2,v(1,0,f+2),dZt/2)
        Fb=dXp*dYp*(w(1,1,f+1)+w(1,1,f))/2
        Ft=dXp*dYp*(w(1,1,f+1)+w(1,1,f+2))/2
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+Ab+At+Apo
        Apw(0,0,f)=Ap
        b(0)=Ap
        c(0)=-Ae
        tempc=velonew(temp(1,1,f+1),dZb/2,temp(1,1,f+2),dZt/2)
c        w(0,1,f+1)=w(1,1,f+1)
c        w(1,0,f+1)=w(1,1,f+1)
        d(0)=Aw*w(0,1,f+1)+As*w(1,0,f+1)+An*w(1,2,f+1)+
     &            At*w(1,1,f+2)+Ab*w(1,1,f)
     &    +dXp*dYp*(Press(0,0,f)-Press(0,0,f+1)) + Apo*wo(1,1,f+1)
     &       +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)


c     for middle elements in the first row
       do 806 k=2,N-1,1
        dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dYp=ycord(1)-ycord(0)
        dXp=xcord(k)-xcord(k-1)
        dXe=(xcord(k+1)+xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dXw=(xcord(k)+Xcord(k-1))/2-(xcord(k-1)+Xcord(k-2))/2
        dYs=(ycord(1)-ycord(0))/2
        dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
        dZt=zcord(f+2)-zcord(f+1)
        dZb=zcord(f+1)-zcord(f)
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
```

```
          Ds=dm1*dXp*dZp/dYs
          Dt=dm1*dXp*dYp/dZt
          Db=dm1*dXp*dYp/dZb
          Fe=dYp*dZp*velonew(u(k,1,f+1),dZb/2,u(k,1,f+2),dZt/2)
          Fw=dYp*dZp*velonew(u(k-1,1,f+1),dZb/2,u(k-1,1,f+2),dZt/2)
          Fn=dXp*dZp*velonew(v(k,1,f+1),dZb/2,v(k,1,f+2),dZt/2)
          Fs=dXp*dZp*velonew(v(k,0,f+1),dZb/2,v(k,0,f+2),dZt/2)
          Fb=dXp*dYp*(w(k,1,f)+w(k,1,f+1))/2
          Ft=dXp*dYp*(w(k,1,f+1)+w(k,1,f+2))/2
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
          Apw(k-1,0,f)=Ap
          a(k-1)=-Aw
          b(k-1)=Ap
          c(k-1)=-Ae
          tempc=velonew(temp(k,1,f+1),dZb/2,temp(k,1,f+2),dZt/2)
c          w(k,0,f+1)=w(k,1,f+1)
          d(k-1)=As*w(k,0,f+1)+An*w(k,2,f+1)+At*w(k,1,f+2)+Ab*w(k,1,f)
     &    +dXp*dYp*(Press(k-1,0,f)-Press(k-1,0,f+1)) + Apo*wo(k,1,f+1)
     &         +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

 806  continue
c     For last element in the first row
        dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dYp=ycord(1)-ycord(0)
        dXp=Xcord(N)-Xcord(N-1)
        dXe=(xcord(N)-Xcord(N-1))/2
        dXw=(xcord(N)+Xcord(N-1))/2-(xcord(N-1)+Xcord(N-2))/2
        dYs=(ycord(1)-ycord(0))/2
        dYn=(ycord(2)+ycord(1))/2-(ycord(1)+ycord(0))/2
        dZt=zcord(f+2)-zcord(f+1)
        dZb=zcord(f+1)-zcord(f)
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(N,1,f+1),dZb/2,u(N,1,f+2),dZt/2)
        Fw=dYp*dZp*velonew(u(N-1,1,f+1),dZb/2,u(N-1,1,f+2),dZt/2)
        Fn=dXp*dZp*velonew(v(N,1,f+1),dZb/2,v(N,1,f+2),dZt/2)
        Fs=dXp*dZp*velonew(v(N,0,f+1),dZb/2,v(N,0,f+2),dZt/2)
        Fb=dXp*dYp*(w(N,1,f)+w(N,1,f+1))/2
        Ft=dXp*dYp*(w(N,1,f+1)+w(N,1,f+2))/2
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
```

```
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      Apw(N-1,0,f)=Ap
      a(N-1)=-Aw
      b(N-1)=Ap
      tempc=velonew(temp(N,1,f+1),dZb/2,temp(N,1,f+2),dZt/2)
c      w(N+1,1,f+1)=w(N,1,f+1)
c      w(N,0,f+1)=w(N,1,f+1)
      d(N-1)=Ae*w(N+1,1,f+1)+As*w(N,0,f+1)+An*w(N,2,f+1)+
     &     At*w(N,1,f+2)+Ab*w(N,1,f) +
     & dXp*dYp*(Press(N-1,0,f)-Press(N-1,0,f+1))+ Apo*wo(N,1,f+1)
     &      +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

c    Call the Thomas Algorithm subprogram
      Call Thomas_algo(N)
      do 906 k=1,N,1
          w(k,1,f+1)=(1.-wrelax)*w(k,1,f+1) + wrelax*x(k)
 906  continue
c    For middle rows
      do 1006 j=1,M-2,1
c    For first cell in the row
      dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dYp=ycord(j+1)-ycord(j)
      dXp=xcord(1)-xcord(0)
      dXe=(xcord(2)+xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYs=(ycord(j+1)+ycord(j))/2-(Ycord(j)+Ycord(j-1))/2
      dYn=(ycord(j+2)+ycord(j+1))/2-(ycord(j+1)+ycord(j))/2
      dZt=zcord(f+2)-zcord(f+1)
      dZb=zcord(f+1)-zcord(f)
      De=dm1*dYp*dZp/dXe
      Dw=dm1*dYp*dZp/dXw
      Dn=dm1*dXp*dZp/dYn
      Ds=dm1*dXp*dZp/dYs
      Dt=dm1*dXp*dYp/dZt
      Db=dm1*dXp*dYp/dZb
      Fe=dYp*dZp*velonew(u(1,j+1,f+1),dZb/2,u(1,j+1,f+2),dZt/2)
      Fw=dYp*dZp*velonew(u(0,j+1,f+1),dZb/2,u(0,j+1,f+2),dZt/2)
      Fn=dXp*dZp*velonew(v(1,j+1,f+1),dZb/2,v(1,j+1,f+2),dZt/2)
      Fs=dXp*dZp*velonew(v(1,j,f+1),dZb/2,v(1,j,f+2),dZt/2)
      Fb=dXp*dYp*(w(1,j+1,f+1)+w(1,j+1,f))/2
      Ft=dXp*dYp*(w(1,j+1,f+1)+w(1,j+1,f+2))/2
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      Apw(0,j,f)=Ap
      b(0)=Ap
      c(0)=-Ae
      tempc=velonew(temp(1,j+1,f+1),dZb/2,temp(1,j+1,f+2),dZt/2)
c      w(0,j+1,f+1)=w(1,j+1,f+1)
      d(0)=Aw*w(0,j+1,f+1)+As*w(1,j,f+1)+An*w(1,j+2,f+1)+
     &       At*w(1,j+1,f+2)+Ab*w(1,j+1,f)+
     &  dXp*dYp*(Press(0,j,f)-Press(0,j,f+1))+Apo*wo(1,j+1,f+1)
     &        +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

c     For middle cells in the row
        do 1106 k=2,N-1,1
          dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
          dYp=ycord(j+1)-ycord(j)
          dXp=xcord(k)-xcord(k-1)
          dXe=(xcord(k+1)+xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
          dXw=(xcord(k)+Xcord(k-1))/2-(xcord(k-1)+Xcord(k-2))/2
          dYs=(ycord(j+1)+ycord(j))/2-(Ycord(j)+Ycord(j-1))/2
          dYn=(ycord(j+2)+ycord(j+1))/2-(ycord(j+1)+ycord(j))/2
          dZt=zcord(f+2)-zcord(f+1)
          dZb=zcord(f+1)-zcord(f)
          De=dm1*dYp*dZp/dXe
          Dw=dm1*dYp*dZp/dXw
          Dn=dm1*dXp*dZp/dYn
          Ds=dm1*dXp*dZp/dYs
          Dt=dm1*dXp*dYp/dZt
          Db=dm1*dXp*dYp/dZb
          Fe=dYp*dZp*velonew(u(k,j+1,f+1),
     &          dZb/2,u(k,j+1,f+2),dZt/2)
          Fw=dYp*dZp*velonew(u(k-1,j+1,f+1),
     &          dZb/2,u(k-1,j+1,f+2),dZt/2)
          Fn=dXp*dZp*velonew(v(k,j+1,f+1),
     &                  dZb/2,v(k,j+1,f+2),dZt/2)
          Fs=dXp*dZp*velonew(v(k,j,f+1),dZb/2,v(k,j,f+2),dZt/2)
          Fb=dXp*dYp*(w(k,j+1,f)+w(k,j+1,f+1))/2
          Ft=dXp*dYp*(w(k,j+1,f+1)+w(k,j+1,f+2))/2
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
          Apw(k-1,j,f)=Ap
          a(k-1)=-Aw
          b(k-1)=Ap
          c(k-1)=-Ae
          tempc=velonew(temp(k,j+1,f+1),dZb/2,temp(k,j+1,f+2),dZt/2)
          d(k-1)=As*w(k,j,f+1)+An*w(k,j+2,f+1)+
     &         At*w(k,j+1,f+2)+Ab*w(k,j+1,f)+
     &  dXp*dYp*(Press(k-1,j,f)-Press(k-1,j,f+1))+ Apo*wo(k,j+1,f+1)
     &      +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

 1106   continue
c     For last cell in the row
        dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dYp=ycord(j+1)-ycord(j)
        dXp=xcord(N)-xcord(N-1)
        dXe=(xcord(N)-Xcord(N-1))/2
        dXw=(xcord(N)+Xcord(N-1))/2-(xcord(N-1)+Xcord(N-2))/2
        dYs=(ycord(j+1)+ycord(j))/2-(Ycord(j)+Ycord(j-1))/2
```

```
         dYn=(ycord(j+2)+ycord(j+1))/2-(ycord(j+1)+ycord(j))/2
         dZt=zcord(f+2)-zcord(f+1)
         dZb=zcord(f+1)-zcord(f)
         De=dm1*dYp*dZp/dXe
         Dw=dm1*dYp*dZp/dXw
         Dn=dm1*dXp*dZp/dYn
         Ds=dm1*dXp*dZp/dYs
         Dt=dm1*dXp*dYp/dZt
         Db=dm1*dXp*dYp/dZb
         Fe=dYp*dZp*velonew(u(N,j+1,f+1),dZb/2,u(N,j+1,f+2),dZt/2)
         Fw=dYp*dZp*velonew(u(N-1,j+1,f+1),
     &         dZb/2,u(N-1,j+1,f+2),dZt/2)
         Fn=dXp*dZp*velonew(v(N,j+1,f+1),
     &         dZb/2,v(N,j+1,f+2),dZt/2)
         Fs=dXp*dZp*velonew(v(N,j,f+1),dZb/2,v(N,j,f+2),dZt/2)
         Fb=dXp*dYp*(w(N,j+1,f)+w(N,j+1,f+1))/2
         Ft=dXp*dYp*(w(N,j+1,f+1)+w(N,j+1,f+2))/2
         Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
         Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
         An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
         As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
         At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
         Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
         Apo=dXp*dYp*dZp/dtime
         Ap=Ae+Aw+An+As+At+Ab+Apo
         Apw(N-1,j,f)=Ap
         a(N-1)=-Aw
         b(N-1)=Ap
         tempc=velonew(temp(N,j+1,f+1),dZb/2,temp(N,j+1,f+2),dZt/2)
c        w(N+1,j+1,f+1)=w(N,j+1,f+1)
        d(N-1)=Ae*w(N+1,j+1,f+1)+As*w(N,j,f+1)+An*w(N,j+2,f+1)+
     &        At*w(N,j+1,f+2)+Ab*w(N,j+1,f)+
     &     dXp*dYp*(Press(N-1,j,f)-Press(N-1,j,f+1))+Apo*wo(N,j+1,f+1)
     &        +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

c     Call the Thomas Algorithm subprogram
        Call Thomas_algo(N)
        do 1206 k=1,N,1
          w(k,j+1,f+1)=(1.-wrelax)*w(k,j+1,f+1) + wrelax*x(k)
1206    continue

1006   continue
c     For the last row of elements
       dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
       dYp=ycord(M)-ycord(M-1)
       dXp=xcord(1)-xcord(0)
       dXe=(xcord(2)+xcord(1))/2-(Xcord(1)+Xcord(0))/2
       dXw=(Xcord(1)-Xcord(0))/2
       dYs=(ycord(M)+ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
       dYn=(ycord(M)-ycord(M-1))/2
       dZt=zcord(f+2)-zcord(f+1)
       dZb=zcord(f+1)-zcord(f)
       De=dm1*dYp*dZp/dXe
       Dw=dm1*dYp*dZp/dXw
       Dn=dm1*dXp*dZp/dYn
       Ds=dm1*dXp*dZp/dYs
```

```
      Dt=dm1*dXp*dYp/dZt
      Db=dm1*dXp*dYp/dZb
      Fe=dYp*dZp*velonew(u(1,M,f+1),dZb/2,u(1,M,f+2),dZt/2)
      Fw=dYp*dZp*velonew(u(0,M,f+1),dZb/2,u(0,M,f+2),dZt/2)
      Fn=dXp*dZp*velonew(v(1,M,f+1),dZb/2,v(1,M,f+2),dZt/2)
      Fs=dXp*dZp*velonew(v(1,M-1,f+1),dZb/2,v(1,M-1,f+2),dZt/2)
      Fb=dXp*dYp*(w(1,M,f)+w(1,M,f+1))/2
      Ft=dXp*dYp*(w(1,M,f+1)+w(1,M,f+2))/2
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      Apw(0,M-1,f)=Ap
      b(0)=Ap
      c(0)=-Ae
      tempc=velonew(temp(1,M,f+1),dZb/2,temp(1,M,f+2),dZt/2)
c       w(0,M,f+1)=w(1,M,f+1)
c       w(1,M+1,f+1)=w(1,M,f+1)
      d(0)=Aw*w(0,M,f+1)+As*w(1,M-1,f+1)+An*w(1,M+1,f+1)+
    &       At*w(1,M,f+2)+Ab*w(1,M,f)+
    &       dXp*dYp*(Press(0,M-1,f)-Press(0,M-1,f+1))+Apo*wo(1,M,f+1)
    &        +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

c     For middle elements in the row
       do 1307 k=2,N-1,1
        dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dYp=ycord(M)-ycord(M-1)
        dXp=xcord(k)-xcord(k-1)
        dXe=(xcord(k+1)+xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dXw=(xcord(k)+Xcord(k-1))/2-(xcord(k-1)+Xcord(k-2))/2
        dYs=(ycord(M)+ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
        dYn=(ycord(M)-ycord(M-1))/2
        dZt=zcord(f+2)-zcord(f+1)
        dZb=zcord(f+1)-zcord(f)
        De=dm1*dYp*dZp/dXe
        Dw=dm1*dYp*dZp/dXw
        Dn=dm1*dXp*dZp/dYn
        Ds=dm1*dXp*dZp/dYs
        Dt=dm1*dXp*dYp/dZt
        Db=dm1*dXp*dYp/dZb
        Fe=dYp*dZp*velonew(u(k,M,f+1),dZb/2,u(k,M,f+2),dZt/2)
        Fw=dYp*dZp*velonew(u(k-1,M,f+1),dZb/2,u(k-1,M,f+2),dZt/2)
        Fn=dXp*dZp*velonew(v(k,M,f+1),dZb/2,v(k,M,f+2),dZt/2)
        Fs=dXp*dZp*velonew(v(k,M-1,f+1),dZb/2,v(k,M-1,f+2),dZt/2)
        Fb=dXp*dYp*(w(k,M,f)+w(k,M,f+1))/2
        Ft=dXp*dYp*(w(k,M,f+1)+w(k,M,f+2))/2
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      Apw(k-1,M-1,f)=Ap
      a(k-1)=-Aw
      b(k-1)=Ap
      c(k-1)=-Ae
      tempc=velonew(temp(k,M,f+1),dZb/2,temp(k,M,f+2),dZt/2)
c      w(k,M+1,f+1)=w(k,M,f+1)
      d(k-1)=As*w(k,M-1,f+1)+An*w(k,M+1,f+1)+
     &       At*w(k,M,f+2)+Ab*w(k,M,f)+
     &   dXp*dYp*(Press(k-1,M-1,f)-Press(k-1,M-1,f+1))+Apo*wo(k,M,f+1)
     &       +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)

 1307 continue
c   For last element in the row
      dZp=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dYp=ycord(M)-ycord(M-1)
      dXp=xcord(N)-xcord(N-1)
      dXe=(xcord(N)-Xcord(N-1))/2
      dXw=(xcord(N)+Xcord(N-1))/2-(xcord(N-1)+Xcord(N-2))/2
      dYs=(ycord(M)+ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dYn=(ycord(M)-ycord(M-1))/2
      dZt=zcord(f+2)-zcord(f+1)
      dZb=zcord(f+1)-zcord(f)
      De=dm1*dYp*dZp/dXe
      Dw=dm1*dYp*dZp/dXw
      Dn=dm1*dXp*dZp/dYn
      Ds=dm1*dXp*dZp/dYs
      Dt=dm1*dXp*dYp/dZt
      Db=dm1*dXp*dYp/dZb
      Fe=dYp*dZp*velonew(u(N,M,f+1),dZb/2,u(N,M,f+2),dZt/2)
      Fw=dYp*dZp*velonew(u(N-1,M,f+1),dZb/2,u(N-1,M,f+2),dZt/2)
      Fn=dXp*dZp*velonew(v(N,M,f+1),dZb/2,v(N,M,f+2),dZt/2)
      Fs=dXp*dZp*velonew(v(N,M-1,f+1),dZb/2,v(N,M-1,f+2),dZt/2)
      Fb=dXp*dYp*(w(N,M,f)+w(N,M,f+1))/2
      Ft=dXp*dYp*(w(N,M,f+1)+w(N,M,f+2))/2
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) +  greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) +  greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) +  greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) +  greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) +  greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) +  greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      Apw(N-1,M-1,f)=Ap
      a(N-1)=-Aw
      b(N-1)=Ap
      tempc=velonew(temp(N,M,f+1),dZb/2,temp(N,M,f+2),dZt/2)
c      w(N+1,M,f+1)=w(N,M,f+1)
c      w(N,M+1,f+1)=w(N,M,f+1)
      d(N-1)=Ae*w(N+1,M,f+1)+As*w(N,M-1,f+1)+An*w(N,M+1,f+1)+
     &       At*w(N,M,f+2) + Ab*w(N,M,f)    +
     &  dXp*dYp*(Press(N-1,M-1,f)-Press(N-1,M-1,f+1))+ Apo*wo(N,M,f+1)
     &       +Re*dm2*(tempc-tini)*dXp*dYp*dZp*cos(Tilt)
```

```
c    Call the Thomas Algorithm subprogram
      Call Thomas_algo(N)
      do 1407 k=1,N,1
         w(k,M,f+1)=(1.-wrelax)*w(k,M,f+1) + wrelax*x(k)
1407  continue
3011 continue


c*************end of w velocity******************************
      werrorn=0.
      werrord=0.
      do 2105 i=0,N+1,1
        do 2205 j=0,M+1,1
          DO 2255 k=0,P,1
             werrorn=werrorn+abs(w(i,j,k)-wold(i,j,k))
             werrord=werrord+abs(w(i,j,k))
2255         continue
2205       continue
2105    continue


     werror=werrorn/werrord

c     write(wc,*)"werror = " ,werror

c    Continue till the velocities converge to a certain limit
     uvwerror=greater(uerror,verror)
     uvwerror=greater(uvwerror,werror)
c     write (wc,*) uvwerror,uerror,verror,werror
3000 continue

c    check if the solution satisfies the continuity equation
     converge=0.0
     do 3200 k=0,P-1,1
     do 3200 j=0,M-1,1
       do 3200 i=0,N-1,1
          dXp=xcord(i+1)-xcord(i)
          dYp=ycord(j+1)-ycord(j)
          dZp=zcord(k+1)-zcord(k)
          barray(i,j,k)=
   &          -(dXp*dZp*(v(i+1,j+1,k+1)-v(i+1,j,k+1))+
   &           dYp*dZp*(u(i+1,j+1,k+1)-u(i,j+1,k+1)) +
   &           dXp*dYp*(w(i+1,j+1,k+1)-w(i+1,j+1,k)))
c    if (converge.lt.abs(barray(i,j,k)))  converge=abs(barray(i,j,k))

          converge=converge + abs(barray(i,j,k))

3200   continue
    converge= converge/(N*M*P)
    if (converge.eq.0.0) converge=1.0
    con1=con1+1
    if (con1.eq.1)  write(wc,*) con1,"  converge= ",converge
    call continuity(Apu,Apv,Apw,press,barray,w,v,u,N,M,P)
3300 continue
    cut=0.0
```

```
      cutd=0.0
      cvt=0.0
      cvtd=0.0
      cwt=0.0
      cwtd=0.0
      do 303 i=0,N,1
        do 303 j=0,M+1,1
            do 303 k=0,P+1,1
              cut=cut + abs(u(i,j,k)-uinitial(i,j,k))
              cutd=cutd + abs(u(i,j,k))
              uinitial(i,j,k)=u(i,j,k)

   303 continue

      do 503 i=0,N+1,1
        do 503 j=0,M,1
          do 503 k=0,P+1,1
            cvt=cvt + abs(v(i,j,k)-vinitial(i,j,k))
            cvtd=cvtd + abs(v(i,j,k))
            vinitial(i,j,k)=v(i,j,k)
   503 continue

      do 457 k=0,P,1
        do 457 j=0,M+1,1
          do 457 i=0,N+1,1
            cwt=cwt + abs(w(i,j,k)-winitial(i,j,k))
            cwtd=cwtd + abs(w(i,j,k))
            winitial(i,j,k)=w(i,j,k)
  457 continue
      if (cutd.eq.0.0) then
               cut= 1.0
             else
               cut=cut/cutd
             end if
      if (cvtd.eq.0.0) then
               cvt= 1.0
             else
               cvt=cvt/cvtd
             end if
      if (cwtd.eq.0.0) then
               cwt= 1.0
             else
               cwt=cwt/cwtd
             end if
      write(wc,*) " To check if the velocity converged"
      write(wc,*) cut,cvt,cwt
      write(wc,*) " Going for temperature "
      Call TempCalc(u,v,w,N,M,P,temp,Pr,dtime,tempo)
10000 continue
      cu=0.
      cv=0.
      cw=0.
      ct=0.0
      cud=0.
      cvd=0.
      cwd=0.
```

```
        ctd=0.0
        do 304 i=0,N,1
           do 304 j=0,M+1,1
                 do 304 k=0,P+1,1
                    cu=cu + abs(u(i,j,k)-uo(i,j,k))
                    cud=cud + abs(u(i,j,k))

     304 continue

        do 305 i=0,N+1,1
           do 305 j=0,M,1
              do 305 k=0,P+1,1
                 cv=cv + abs(v(i,j,k)-vo(i,j,k))
                 cvd=cvd + abs(v(i,j,k))
     305 continue

        do 306 k=0,P,1
           do 306 j=0,M+1,1
              do 306 i=0,N+1,1
                 cw=cw + abs(w(i,j,k)-wo(i,j,k))
                 cwd=cwd + abs(w(i,j,k))

     306 continue
        do 307 k=0,P+1,1
           do 307 j=0,M+1,1
              do 307 i=0,N+1,1
                 ct=ct + abs(Temp(i,j,k)-Tempo(i,j,k))
                 ctd=ctd + abs(Temp(i,j,k))

     307 continue
        if (cud.eq.0.0) then
                     cu=1.0
                  else
                     cu=cu/cud
                  end if

        if (cvd.eq.0.0) then
                     cv=1.0
                  else
                     cv=cv/cvd
                  end if
        if (cwd.eq.0.0) then
                     cw=1.0
                  else
                     cw=cw/cwd
                  end if
        if (ctd.eq.0.0) then
                     ct=1.0
                  else
                     ct=ct/ctd
                  end if
        write(wc,*)
       &     "convegence of velocities and temperature- ",cu,cv,cw,ct
c           dtime = dtime *2
c           if (dtime.gt.0.03) dtime = 0.03
```

```
  write(4,*)timef,u(10,10,5),v(10,10,5),w(10,10,5),temp(10,10,5)

        if (timestep.eq.1)Then
        open (unit=100,file='data1.dat',status='REPLACE')
        write(100,*)' Variables="x","y","z","u","v","w","t","p" '
  write(100,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do k=0,P-1,1
  do j=0,M-1,1
     do i=0,N-1,1
   WRITE(100,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &              (Zcord(K)+Zcord(K+1))/2,
 &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &              temp(I+1,J+1,K+1),press(I,J,K)

        ENDDO
        ENDDO
        ENDDO
        close(100)
        endif

        If (timestep.eq.2)Then
        open (unit=101,file='data2.dat',status='REPLACE')
        write(101,*)' Variables="x","y","z","u","v","w","t","p" '
  write(101,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 4000 k=0,P-1,1
  do 4001 j=0,M-1,1
     do 4002 i=0,N-1,1
    WRITE(101,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &              (Zcord(K)+Zcord(K+1))/2,
 &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &              temp(I+1,J+1,K+1),press(I,J,K)

4002    continue
4001 CONTINUE
4000 continue
        close(101)
        endif

        If (timestep.eq.3)Then
        open (unit=103,file='data3.dat',status='REPLACE')
        write(103,*)' Variables="x","y","z","u","v","w","t","p" '
  write(103,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 55003 k=0,P-1,1
  do 4004 j=0,M-1,1
     do 4005 i=0,N-1,1
  WRITE(103,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &              (Zcord(K)+Zcord(K+1))/2,
 &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &              temp(I+1,J+1,K+1),press(I,J,K)
 4005    continue
 4004 CONTINUE
55003 continue
          close(103)
          endif

     If (timestep.eq.4)Then
          open (unit=104,file='data4.dat',status='REPLACE')
          write(104,*)' Variables="x","y","z","u","v","w","t","p" '
     write(104,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
     do 1963 k=0,P-1,1
     do 1973 j=0,M-1,1
        do 4408 i=0,N-1,1
      WRITE(104,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
    &              (Zcord(K)+Zcord(K+1))/2,
    &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
    &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
    &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
    &              temp(I+1,J+1,K+1),press(I,J,K)

4408    continue
1973 CONTINUE
1963 continue
         close(104)
         endif


         If (timestep.eq.5)Then
         open (unit=105,file='data5.dat',status='REPLACE')
         write(105,*)' Variables="x","y","z","u","v","w","t","p" '
    write(105,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
    do 4009 k=0,P-1,1
    do 4010 j=0,M-1,1
       do 4011 i=0,N-1,1
    WRITE(105,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
   &              (Zcord(K)+Zcord(K+1))/2,
   &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
   &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
   &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
   &              temp(I+1,J+1,K+1),press(I,J,K)

4011    continue
4010 CONTINUE
4009 continue
         close(105)
         endif



         If (timestep.eq.6)Then
         open (unit=106,file='data6.dat',status='REPLACE')
         write(106,*)' Variables="x","y","z","u","v","w","t","p" '
   write(106,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4012 k=0,P-1,1
   do 4013 j=0,M-1,1
      do 4014 i=0,N-1,1
```

```
  WRITE(106,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &               (Zcord(K)+Zcord(K+1))/2,
 &               (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &               (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &               (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &               temp(I+1,J+1,K+1),press(I,J,K)

4014    continue
4013 CONTINUE
4012 continue
        close(106)
        endif

        If (timestep.eq.7)Then
        open (unit=107,file='data7.dat',status='REPLACE')
        write(107,*)' Variables="x","y","z","u","v","w","t","p" '
  write(107,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 4015 k=0,P-1,1
  do 4016 j=0,M-1,1
     do 4017 i=0,N-1,1
  WRITE(107,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &               (Zcord(K)+Zcord(K+1))/2,
 &               (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &               (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &               (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &               temp(I+1,J+1,K+1),press(I,J,K)

4017    continue
4016 CONTINUE
4015 continue
        close(107)
        endif

        If (timestep.eq.8)Then
        open (unit=108,file='data8.dat',status='REPLACE')
        write(108,*)' Variables="x","y","z","u","v","w","t","p" '
  write(108,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 4018 k=0,P-1,1
  do 4019 j=0,M-1,1
     do 4020 i=0,N-1,1
  WRITE(108,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &               (Zcord(K)+Zcord(K+1))/2,
 &               (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &               (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &               (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &               temp(I+1,J+1,K+1),press(I,J,K)

4020    continue
4019 CONTINUE
4018 continue
        CLOSE(108)
        endif
        If (timestep.eq.9)Then
        open (unit=109,file='data9.dat',status='REPLACE')
        write(109,*)' Variables="x","y","z","u","v","w","t","p" '
   write(109,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4021 k=0,P-1,1
   do 4022 j=0,M-1,1
     do 4023 i=0,N-1,1
    WRITE(109,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4023    continue
4022 CONTINUE
4021 continue
        CLOSE(109)
        endif




        If (timestep.eq.10)Then
        open (unit=110,file='data10.dat',status='REPLACE')
        write(110,*)' Variables="x","y","z","u","v","w","t","p" '
   write(110,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4024 k=0,P-1,1
   do 4025 j=0,M-1,1
     do 4026 i=0,N-1,1
   WRITE(110,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4026    continue
4025 CONTINUE
4024 continue
        close(110)
        endif




        If (timestep.eq.11)Then
        open (unit=111,file='data11.dat',status='REPLACE')
        write(111,*)' Variables="x","y","z","u","v","w","t","p" '
   write(111,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4027 k=0,P-1,1
   do 4028 j=0,M-1,1
     do 4029 i=0,N-1,1
   WRITE(111,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &            (Zcord(K)+Zcord(K+1))/2,
  &            (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &            (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &            (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &            temp(I+1,J+1,K+1),press(I,J,K)

4029   continue
4028 CONTINUE
4027 continue
       close(111)
       endif




       If (timestep.eq.12)Then
       open (unit=112,file='data12.dat',status='REPLACE')
       write(112,*)' Variables="x","y","z","u","v","w","t","p" '
  write(112,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 4030 k=0,P-1,1
  do 4031 j=0,M-1,1
    do 4032 i=0,N-1,1
  WRITE(112,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &            (Zcord(K)+Zcord(K+1))/2,
  &            (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &            (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &            (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &            temp(I+1,J+1,K+1),press(I,J,K)

4032   continue
4031 CONTINUE
4030 continue
       close(112)
       endif



       If (timestep.eq.13)Then
       open (unit=113,file='data13.dat',status='REPLACE')
       write(113,*)' Variables="x","y","z","u","v","w","t","p" '
  write(113,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
  do 4033 k=0,P-1,1
  do 4034 j=0,M-1,1
    do 4035 i=0,N-1,1
   WRITE(113,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &            (Zcord(K)+Zcord(K+1))/2,
  &            (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &            (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &            (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &            temp(I+1,J+1,K+1),press(I,J,K)

4035   continue
4034 CONTINUE
4033 continue
       close(113)
```

```
        endif



        If (timestep.eq.14)Then
        open (unit=114,file='data14.dat',status='REPLACE')
        write(114,*)' Variables="x","y","z","u","v","w","t","p" '
   write(114,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4036 k=0,P-1,1
   do 4037 j=0,M-1,1
     do 4038 i=0,N-1,1
   WRITE(114,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &              (Zcord(K)+Zcord(K+1))/2,
  &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &              temp(I+1,J+1,K+1),press(I,J,K)

4038    continue
4037 CONTINUE
4036 continue
        close(114)
        endif



        If (timestep.eq.15)Then
        open (unit=115,file='data15.dat',status='REPLACE')
        write(115,*)' Variables="x","y","z","u","v","w","t","p" '
   write(115,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4039 k=0,P-1,1
   do 4040 j=0,M-1,1
     do 4041 i=0,N-1,1
   WRITE(115,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &              (Zcord(K)+Zcord(K+1))/2,
  &              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &              temp(I+1,J+1,K+1),press(I,J,K)

4041    continue
4040 CONTINUE
4039 continue
        close(115)
        endif


        If (timestep.eq.16)Then
        open (unit=116,file='data16.dat',status='REPLACE')
        write(116,*)' Variables="x","y","z","u","v","w","t","p" '
   write(116,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4042 k=0,P-1,1
   do 4043 j=0,M-1,1
     do 4044 i=0,N-1,1
   WRITE(116,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4044    continue
4043 CONTINUE
4042 continue
        close(116)
        endif



        If (timestep.eq.17)Then
        open (unit=117,file='data17.dat',status='REPLACE')
        write(117,*)' Variables="x","y","z","u","v","w","t","p" '
   write(117,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4045 k=0,P-1,1
   do 4046 j=0,M-1,1
     do 4047 i=0,N-1,1
    WRITE(117,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4047    continue
4046 CONTINUE
4045 continue
        close(117)
        endif

        If (timestep.eq.18)Then
        open (unit=118,file='data18.dat',status='REPLACE')
        write(118,*)' Variables="x","y","z","u","v","w","t","p" '
   write(118,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4048 k=0,P-1,1
   do 4049 j=0,M-1,1
     do 4050 i=0,N-1,1
   WRITE(118,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4050    continue
4049 CONTINUE
4048 continue
        close(118)
        endif
        If(timestep.eq.19)Then
        open (unit=119,file='data19.dat',status='REPLACE')
        write(119,*)' Variables="x","y","z","u","v","w","t","p" '
   write(119,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4051 k=0,P-1,1
   do 4052 j=0,M-1,1
     do 4053 i=0,N-1,1
   WRITE(119,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4053    continue
4052 CONTINUE
4051 continue
        close(119)
        endif



        IF(timestep.eq.20)Then
        open (unit=120,file='data20.dat',status='REPLACE')
        write(120,*)' Variables="x","y","z","u","v","w","t","p" '
   write(120,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 4054 k=0,P-1,1
   do 4055 j=0,M-1,1
     do 4056 i=0,N-1,1
   WRITE(120,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)

4056    continue
4055 CONTINUE
4054 continue
        close(120)
        endif


        If (timestep.eq.21)Then
        open (unit=121,file='data21.dat',status='REPLACE')
        write(121,*)' Variables="x","y","z","u","v","w","t","p" '
   write(121,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do k=0,P-1,1
   do j=0,M-1,1
     do i=0,N-1,1
    WRITE(121,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
  &             (Zcord(K)+Zcord(K+1))/2,
  &             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
  &             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
  &             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
  &             temp(I+1,J+1,K+1),press(I,J,K)
```

```
      enddo
      enddo
      enddo
      close(121)
      endif




      If (timestep.eq.22)Then
      open (unit=122,file='data22.dat',status='REPLACE')
      write(122,*)' Variables="x","y","z","u","v","w","t","p" '
 write(122,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(122,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&             (Zcord(K)+Zcord(K+1))/2,
&             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&             temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(122)
      endif



      If (timestep.eq.23)Then
      open (unit=123,file='data23.dat',status='REPLACE')
      write(123,*)' Variables="x","y","z","u","v","w","t","p" '
 write(123,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
  WRITE(123,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&             (Zcord(K)+Zcord(K+1))/2,
&             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&             temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(123)
      endif
      If (timestep.eq.24)Then
      open (unit=124,file='data24.dat',status='REPLACE')
      write(124,*)' Variables="x","y","z","u","v","w","t","p" '
 write(124,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(124,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&             (Zcord(K)+Zcord(K+1))/2,
&             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&             temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(124)
      endif



      If (timestep.eq.25)Then
      open (unit=125,file='data25.dat',status='REPLACE')
      write(125,*)' Variables="x","y","z","u","v","w","t","p" '
 write(125,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(125,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&             (Zcord(K)+Zcord(K+1))/2,
&             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&             temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(125)
      endif


      If (timestep.eq.26)Then
      open (unit=126,file='data26.dat',status='REPLACE')
      write(126,*)' Variables="x","y","z","u","v","w","t","p" '
 write(126,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(126,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&             (Zcord(K)+Zcord(K+1))/2,
&             (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&             (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&             (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &                temp(I+1,J+1,K+1),press(I,J,K)

        enddo
        enddo
        enddo
        close(126)
        endif



        If (timestep.eq.27)Then
        open (unit=127,file='data27.dat',status='REPLACE')
        write(127,*)' Variables="x","y","z","u","v","w","t","p" '
 write(127,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
  WRITE(127,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
 &               (Zcord(K)+Zcord(K+1))/2,
 &               (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
 &               (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
 &               (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
 &               temp(I+1,J+1,K+1),press(I,J,K)

        enddo
        enddo
        enddo
        close(127)
        endif

        If (timestep.eq.28)Then
        open (unit=128,file='data28.dat',status='REPLACE')
        write(128,*)' Variables="x","y","z","u","v","w","t","p" '
 write(128,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(128,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&               (Zcord(K)+Zcord(K+1))/2,
&               (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&               (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&               (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&               temp(I+1,J+1,K+1),press(I,J,K)

        enddo
        enddo
        enddo
        close(128)
        endif


        If(timestep.eq.29)Then
        open (unit=129,file='data29.dat',status='REPLACE')
        write(129,*)' Variables="x","y","z","u","v","w","t","p" '
write(129,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
do k=0,P-1,1
```

```
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(129,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&              (Zcord(K)+Zcord(K+1))/2,
&              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&              temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(129)
      endif



      IF(timestep.eq.30)Then
      open (unit=130,file='data30.dat',status='REPLACE')
      write(130,*)' Variables="x","y","z","u","v","w","t","p" '
 write(130,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
 do k=0,P-1,1
 do j=0,M-1,1
   do i=0,N-1,1
 WRITE(130,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&              (Zcord(K)+Zcord(K+1))/2,
&              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
&              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
&              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
&              temp(I+1,J+1,K+1),press(I,J,K)

      enddo
      enddo
      enddo
      close(130)



      EndIf


      TimeStep=TimeStep+1
15000 continue


   open (unit=2,file='outvT.dat',status='NEW')
   open (unit=1,file='data.dat',status='NEW')
   write(2,*) u,v,w,Temp,Press
   write(1,*) "ZONE I=",N,", J=",M,", K=",P,", F=POINT"
   do 3303 k=0,P-1,1
   do 3302 j=0,M-1,1
     do 3301 i=0,N-1,1
       WRITE(1,*) (XCORD(I)+XCORD(I+1))/2,(YCORD(j)+YCORD(j+1))/2,
&              (Zcord(K)+Zcord(K+1))/2,
&              (U(I,J+1,K+1)+U(I+1,J+1,K+1))/2,
     &              (V(I+1,J,K+1)+V(I+1,J+1,K+1))/2,
     &              (w(I+1,J+1,K+1)+w(I+1,J+1,K))/2,
     &              temp(I+1,J+1,K+1),press(I,J,K)

 3301    continue
 3302 CONTINUE
 3303 continue

      CLOSE(UNIT=1)


c     End of main program
      end


c     Subprogram for correcting the pressure or solving the pprime eqn
c     Construct the required matrices for line iteration using thomas algo
c     for first row of elements

      Subroutine Continuity(Apu,Apv,Apw,press,barray,w,v,u,N,M,P)
      Common/grid/Xcord(0:40),Ycord(0:40),Zcord(0:40)
      integer N,M,P
      real a,b,c,d,x
      real prelax
      Common/thomas/ a(0:40),b(0:40),c(0:40),d(0:40),x(1:41)
      real u(0:N,0:M+1,0:P+1)
      real v(0:N+1,0:M,0:P+1)
      real w(0:N+1,0:M+1,0:P)
      real error,errorn,errord
      real ap,aw,ae,an,as,at,ab
      real Apu(0:N-1,0:M-1,0:P-1)
      real Apv(0:N-1,0:M-1,0:P-1)
      real Apw(0:N-1,0:M-1,0:P-1)
      real press(0:N-1,0:M-1,0:P-1)
      real barray(0:N-1,0:M-1,0:P-1)
      integer pcon
      real pp(0:40,0:40,0:40),ppold(0:40,0:40,0:40)
      real avgpress,pcorrect,ppavg
      prelax=0.8
      do 310 k=0,P-1,1
       do 310 i=0,N-1,1
        do 310 j=0,M-1,1
           pp(i,j,k)=0
 310    continue
      error=1.
      pcon=0
      do 1400 while (error.gt.0.0001.and.pcon.lt.10)
      pcon=pcon+1
      do 300 k=0,P-1,1
       do 300 i=0,N-1,1
        do 300 j=0,M-1,1
           ppold(i,j,k)=pp(i,j,k)
 300    continue

      dXp=Xcord(1)-Xcord(0)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(1)-Zcord(0)
      Ae=dYp*dZp*dYp*dZp/Apu(0,0,0)
      Aw=0
      An=dXp*dZp*dXp*dZp/Apv(0,0,0)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(0,0,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      b(0)=Ap
      c(0)=-Ae
      d(0)=An*pp(0,1,0)+ At*pp(0,0,1)+barray(0,0,0)


      do 500 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(1)-Zcord(0)
      Ae=dYp*dZp*dYp*dZp/Apu(k,0,0)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,0,0)
      An=dXp*dZp*dXp*dZp/Apv(k,0,0)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(k,0,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=An*pp(k,1,0) + At*pp(k,0,1)+ barray(k,0,0)

  500 continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=zcord(1)-zcord(0)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,0,0)
      An=dXp*dZp*dXp*dZp/Apv(N-1,0,0)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(N-1,0,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=An*pp(N-1,1,0) + At*pp(N-1,0,1) + barray(N-1,0,0)

c     call thomas algorithm
      call Thomas_algo(N)
      do 600 k=0,N-1,1
      pp(k,0,0)=(1.-prelax)*pp(k,0,0) + prelax*x(k+1)
  600 continue
c     for middle row of elements

      do 800 j=1,M-2,1
        dXp=xcord(1)-xcord(0)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(1)-Zcord(0)
        Ae=dYp*dZp*dYp*dZp/Apu(0,j,0)
        Aw=0
        An=dXp*dZp*dXp*dZp/Apv(0,j,0)
        As=dXp*dZp*dXp*dZp/Apv(0,j-1,0)
        At=dXp*dYp*dXp*dYp/Apw(0,j,0)
        Ab=0
        Ap=Ae+Aw+An+As+At+Ab
        b(0)=Ap
        c(0)=-Ae
        d(0)=An*pp(0,j+1,0)+ As*pp(0,j-1,0)+At*pp(0,j,1) +
     &    barray(0,j,0)

        do 700 k=1,N-2,1
           dXp=xcord(k+1)-xcord(k)
           dYp=ycord(j+1)-ycord(j)
           dZp=Zcord(1)-Zcord(0)
           Ae=dYp*dZp*dYp*dZp/Apu(k,j,0)
           Aw=dYp*dZp*dYp*dZp/Apu(k-1,j,0)
           An=dXp*dZp*dXp*dZp/Apv(k,j,0)
           As=dXp*dZp*dXp*dZp/Apv(k,j-1,0)
           At=dXp*dYp*dXp*dYp/Apw(k,j,0)
           Ab=0
           Ap=Ae+Aw+An+As+At+Ab
           a(k)=-Aw
           b(k)=Ap
           c(k)=-Ae
           d(k)=An*pp(k,j+1,0)+ As*pp(k,j-1,0) + At*pp(k,j,1)+
     &          barray(k,j,0)
 700       continue

        dXp=xcord(N)-xcord(N-1)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(1)-Zcord(0)
        Ae=0
        Aw=dYp*dZp*dYp*dZp/Apu(N-2,j,0)
        An=dXp*dZp*dXp*dZp/Apv(N-1,j,0)
        As=dXp*dZp*dXp*dZp/Apv(N-1,j-1,0)
        At=dXp*dYp*dXp*dYp/Apw(N-1,j,0)
        Ab=0
        Ap=Ae+Aw+An+As+At+Ab
        a(N-1)=-Aw
        b(N-1)=Ap
        d(N-1)=An*pp(N-1,j+1,0)+ As*pp(N-1,j-1,0) + At*pp(N-1,j,1) +
     &    barray(N-1,j,0)
c     call thomas algorithm
      call Thomas_algo(N)
      do 900 k=0,N-1,1
      pp(k,j,0)=(1.-prelax)*pp(k,j,0) + prelax*x(k+1)
 900 continue

 800 continue
c     for last row of elements
      dXp=xcord(1)-xcord(0)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(1)-Zcord(0)
      Ae=dYp*dZp*dYp*dZp/Apu(0,M-1,0)
      Aw=0
      An=0
      As=dXp*dZp*dXp*dZp/Apv(0,M-2,0)
      At=dXp*dYp*dXp*dYp/Apw(0,M-1,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      b(0)=Ap
      c(0)=-Ae
      d(0)=As*pp(0,M-2,0)+ At*pp(0,M-1,1) +
     &      barray(0,M-1,0)

      do 1000 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(1)-Zcord(0)
      Ae=dYp*dZp*dYp*dZp/Apu(k,M-1,0)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,M-1,0)
      An=0
      As=dXp*dZp*dXp*dZp/Apv(k,M-2,0)
      At=dXp*dYp*dXp*dYp/Apw(k,M-1,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=As*pp(k,M-2,0) + At*pp(k,M-1,1) +
     &      barray(k,M-1,0)
 1000 continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(1)-Zcord(0)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,M-1,0)
      As=dXp*dZp*dXp*dZp/Apv(N-1,M-2,0)
      An=0
      At=dXp*dYp*dXp*dYp/Apw(N-1,M-1,0)
      Ab=0
      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=As*pp(N-1,M-2,0) + At*pp(N-1,M-1,1) +
     &   barray(N-1,M-1,0)
c     call thomas algorithm
      call Thomas_algo(N)
      do 1100 k=0,N-1,1
      pp(k,M-1,0)=(1.-prelax)*pp(k,M-1,0) + prelax*x(k+1)
 1100 continue
c     For middle planes
      do 1123 f=1,P-2,1
      dXp=Xcord(1)-Xcord(0)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(f+1)-Zcord(f)
      Ae=dYp*dZp*dYp*dZp/Apu(0,0,f)
      Aw=0
      An=dXp*dZp*dXp*dZp/Apv(0,0,f)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(0,0,f)
      Ab=dXp*dYp*dXp*dYp/Apw(0,0,f-1)
      Ap=Ae+Aw+An+As+At+Ab
      b(0)=Ap
      c(0)=-Ae
      d(0)=An*pp(0,1,f)+ At*pp(0,0,f+1)+ Ab*pp(0,0,f-1)+
     &   barray(0,0,f)

      do 502 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(f+1)-Zcord(f)
      Ae=dYp*dZp*dYp*dZp/Apu(k,0,f)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,0,f)
      An=dXp*dZp*dXp*dZp/Apv(k,0,f)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(k,0,f)
      Ab=dXp*dYp*dXp*dYp/Apw(k,0,f-1)
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=An*pp(k,1,f) + At*pp(k,0,f+1)+ Ab*pp(k,0,f-1) +
     &   barray(k,0,f)
  502 continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=zcord(f+1)-zcord(f)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,0,f)
      An=dXp*dZp*dXp*dZp/Apv(N-1,0,f)
      As=0
      At=dXp*dYp*dXp*dYp/Apw(N-1,0,f)
      Ab=dXp*dYp*dXp*dYp/Apw(N-1,0,f-1)

      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=An*pp(N-1,1,f) + At*pp(N-1,0,f+1) + Ab*pp(N-1,0,f-1) +
     &       barray(N-1,0,f)
c     call thomas algorithm
      call Thomas_algo(N)
      do 602 k=0,N-1,1
      pp(k,0,f)=(1.-prelax)*pp(k,0,f) + prelax*x(k+1)
  602 continue
c     for middle row of elements

      do 802 j=1,M-2,1
         dXp=xcord(1)-xcord(0)
         dYp=ycord(j+1)-ycord(j)
         dZp=Zcord(f+1)-Zcord(f)
         Ae=dYp*dZp*dYp*dZp/Apu(0,j,f)
         Aw=0
         An=dXp*dZp*dXp*dZp/Apv(0,j,f)
         As=dXp*dZp*dXp*dZp/Apv(0,j-1,f)
         At=dXp*dYp*dXp*dYp/Apw(0,j,f)
         Ab=dXp*dYp*dXp*dYp/Apw(0,j,f-1)
```

```
        Ap=Ae+Aw+An+As+At+Ab
        b(0)=Ap
        c(0)=-Ae
        d(0)=An*pp(0,j+1,f)+ As*pp(0,j-1,f)+ Ab*pp(0,j,f-1)  +
     &    At*pp(0,j,f+1)+barray(0,j,f)

        do 702 k=1,N-2,1
           dXp=xcord(k+1)-xcord(k)
           dYp=ycord(j+1)-ycord(j)
           dZp=Zcord(f+1)-Zcord(f)
           Ae=dYp*dZp*dYp*dZp/Apu(k,j,f)
           Aw=dYp*dZp*dYp*dZp/Apu(k-1,j,f)
           An=dXp*dZp*dXp*dZp/Apv(k,j,f)
           As=dXp*dZp*dXp*dZp/Apv(k,j-1,f)
           At=dXp*dYp*dXp*dYp/Apw(k,j,f)
           Ab=dXp*dYp*dXp*dYp/Apw(k,j,f-1)
           Ap=Ae+Aw+An+As+At+Ab
           a(k)=-Aw
           b(k)=Ap
           c(k)=-Ae
           d(k)=An*pp(k,j+1,f)+ As*pp(k,j-1,f) + At*pp(k,j,f+1)+
     &          Ab*pp(k,j,f-1) + barray(k,j,f)
  702      continue

        dXp=xcord(N)-xcord(N-1)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(f+1)-Zcord(f)
        Ae=0
        Aw=dYp*dZp*dYp*dZp/Apu(N-2,j,f)
        An=dXp*dZp*dXp*dZp/Apv(N-1,j,f)
        As=dXp*dZp*dXp*dZp/Apv(N-1,j-1,f)
        At=dXp*dYp*dXp*dYp/Apw(N-1,j,f)
        Ab=dXp*dYp*dXp*dYp/Apw(N-1,j,f-1)
        Ap=Ae+Aw+An+As+At+Ab
        a(N-1)=-Aw
        b(N-1)=Ap
        d(N-1)=An*pp(N-1,j+1,f)+ As*pp(N-1,j-1,f) + At*pp(N-1,j,f+1) +
     &    Ab*pp(N-1,j,f-1) + barray(N-1,j,f)
c     call thomas algorithm
      call Thomas_algo(N)
      do 902 k=0,N-1,1
      pp(k,j,f)=(1.-prelax)*pp(k,j,f) + prelax*x(k+1)
  902 continue

  802 continue
c     for last row of elements
      dXp=xcord(1)-xcord(0)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(f+1)-Zcord(f)
      Ae=dYp*dZp*dYp*dZp/Apu(0,M-1,f)
      Aw=0
      An=0
      As=dXp*dZp*dXp*dZp/Apv(0,M-2,f)
      At=dXp*dYp*dXp*dYp/Apw(0,M-1,f)
      Ab=dXp*dYp*dXp*dYp/Apw(0,M-1,f-1)
      Ap=Ae+Aw+An+As+At+Ab
      b(0)=Ap
      c(0)=-Ae
      d(0)=As*pp(0,M-2,f)+ At*pp(0,M-1,f+1) +Ab*pp(0,M-1,f-1) +
     &      barray(0,M-1,f)

      do 1002 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(f+1)-Zcord(f)
      Ae=dYp*dZp*dYp*dZp/Apu(k,M-1,f)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,M-1,f)
      An=0
      As=dXp*dZp*dXp*dZp/Apv(k,M-2,f)
      At=dXp*dYp*dXp*dYp/Apw(k,M-1,f)
      Ab=dXp*dYp*dXp*dYp/Apw(k,M-1,f-1)
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=As*pp(k,M-2,f) + At*pp(k,M-1,f+1) +Ab*pp(k,M-1,f-1) +
     &      barray(k,M-1,f)
 1002 continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(f+1)-Zcord(f)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,M-1,f)
      As=dXp*dZp*dXp*dZp/Apv(N-1,M-2,f)
      An=0
      At=dXp*dYp*dXp*dYp/Apw(N-1,M-1,f)
      Ab=dXp*dYp*dXp*dYp/Apw(N-1,M-1,f-1)
      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=As*pp(N-1,M-2,f) + At*pp(N-1,M-1,f+1) +Ab*pp(N-1,M-1,f-1) +
     &  barray(N-1,M-1,f)
c     call thomas algorithm
      call Thomas_algo(N)
      do 1102 k=0,N-1,1
      pp(k,M-1,f)=(1.-prelax)*pp(k,M-1,f) + prelax*x(k+1)
 1102 continue

 1123 Continue
c     For the top most plane
      dXp=Xcord(1)-Xcord(0)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(P)-Zcord(P-1)
      Ae=dYp*dZp*dYp*dZp/Apu(0,0,P-1)
      Aw=0
      An=dXp*dZp*dXp*dZp/Apv(0,0,P-1)
      As=0
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(0,0,P-2)
      Ap=Ae+Aw+An+As+At+Ab
      b(0)=Ap
      c(0)=-Ae
      d(0)=An*pp(0,1,P-1)+  Ab*pp(0,0,P-2)+
     &   barray(0,0,P-1)

      do 507 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(P)-Zcord(P-1)
      Ae=dYp*dZp*dYp*dZp/Apu(k,0,P-1)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,0,P-1)
      An=dXp*dZp*dXp*dZp/Apv(k,0,P-1)
      As=0
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(k,0,P-2)
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=An*pp(k,1,P-1) + Ab*pp(k,0,P-2) +
     &    barray(k,0,P-1)
 507  continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=zcord(P)-zcord(P-1)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,0,P-1)
      An=dXp*dZp*dXp*dZp/Apv(N-1,0,P-1)
      As=0
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(N-1,0,P-2)

      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=An*pp(N-1,1,P-1) +  Ab*pp(N-1,0,P-2) +
     &       barray(N-1,0,P-1)
c     call thomas algorithm
      call Thomas_algo(N)
      do 607 k=0,N-1,1
      pp(k,0,P-1)=(1.-prelax)*pp(k,0,P-1) + prelax*x(k+1)
 607  continue
c     for middle row of elements

      do 807 j=1,M-2,1
        dXp=xcord(1)-xcord(0)
        dYp=ycord(j+1)-ycord(j)
        dZp=zcord(P)-zcord(P-1)
        Ae=dYp*dZp*dYp*dZp/Apu(0,j,P-1)
        Aw=0
        An=dXp*dZp*dXp*dZp/Apv(0,j,P-1)
        As=dXp*dZp*dXp*dZp/Apv(0,j-1,P-1)
        At=0
        Ab=dXp*dYp*dXp*dYp/Apw(0,j,P-2)
        Ap=Ae+Aw+An+As+At+Ab
        b(0)=Ap
        c(0)=-Ae
        d(0)=An*pp(0,j+1,P-1)+ As*pp(0,j-1,P-1)+Ab*pp(0,j,P-2) +
  &   barray(0,j,P-1)

      do 707 k=1,N-2,1
         dXp=xcord(k+1)-xcord(k)
         dYp=ycord(j+1)-ycord(j)
         dZp=Zcord(P)-Zcord(P-1)
         Ae=dYp*dZp*dYp*dZp/Apu(k,j,P-1)
         Aw=dYp*dZp*dYp*dZp/Apu(k-1,j,P-1)
         An=dXp*dZp*dXp*dZp/Apv(k,j,P-1)
         As=dXp*dZp*dXp*dZp/Apv(k,j-1,P-1)
         At=0
         Ab=dXp*dYp*dXp*dYp/Apw(k,j,P-2)
         Ap=Ae+Aw+An+As+At+Ab
         a(k)=-Aw
         b(k)=Ap
         c(k)=-Ae
         d(k)=An*pp(k,j+1,P-1)+ As*pp(k,j-1,P-1) +
  &       Ab*pp(k,j,P-2) + barray(k,j,P-1)
 707     continue

      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(j+1)-ycord(j)
      dZp=Zcord(P)-Zcord(P-1)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,j,P-1)
      An=dXp*dZp*dXp*dZp/Apv(N-1,j,P-1)
      As=dXp*dZp*dXp*dZp/Apv(N-1,j-1,P-1)
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(N-1,j,P-2)
      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=An*pp(N-1,j+1,P-1)+ As*pp(N-1,j-1,P-1) +
    & Ab*pp(N-1,j,P-2) + barray(N-1,j,P-1)
c    call thomas algorithm
     call Thomas_algo(N)
     do 907 k=0,N-1,1
     pp(k,j,P-1)=(1.-prelax)*pp(k,j,P-1) + prelax*x(k+1)
 907 continue

 807 continue
c    for last row of elements
     dXp=xcord(1)-xcord(0)
     dYp=ycord(M)-ycord(M-1)
     dZp=Zcord(P)-Zcord(P-1)
     Ae=dYp*dZp*dYp*dZp/Apu(0,M-1,P-1)
     Aw=0
     An=0
     As=dXp*dZp*dXp*dZp/Apv(0,M-2,P-1)
     At=0
     Ab=dXp*dYp*dXp*dYp/Apw(0,M-1,P-2)
     Ap=Ae+Aw+An+As+At+Ab
     b(0)=Ap
     c(0)=-Ae
     d(0)=As*pp(0,M-2,P-1) +Ab*pp(0,M-1,P-2) +
    &      barray(0,M-1,P-1)
```

```
      do 1007 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(P)-Zcord(P-1)
      Ae=dYp*dZp*dYp*dZp/Apu(k,M-1,P-1)
      Aw=dYp*dZp*dYp*dZp/Apu(k-1,M-1,P-1)
      An=0
      As=dXp*dZp*dXp*dZp/Apv(k,M-2,P-1)
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(k,M-1,P-2)
      Ap=Ae+Aw+An+As+At+Ab
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
      d(k)=As*pp(k,M-2,P-1)  +Ab*pp(k,M-1,P-2) +
     &      barray(k,M-1,P-1)
 1007 continue
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(P)-Zcord(P-1)
      Ae=0
      Aw=dYp*dZp*dYp*dZp/Apu(N-2,M-1,P-1)
      As=dXp*dZp*dXp*dZp/Apv(N-1,M-2,P-1)
      An=0
      At=0
      Ab=dXp*dYp*dXp*dYp/Apw(N-1,M-1,P-2)
      Ap=Ae+Aw+An+As+At+Ab
      a(N-1)=-Aw
      b(N-1)=Ap
      d(N-1)=As*pp(N-1,M-2,P-1) +Ab*pp(N-1,M-1,P-2) +
     &   barray(N-1,M-1,P-1)
c     call thomas algorithm
      call Thomas_algo(N)
      do 1107 k=0,N-1,1
      pp(k,M-1,P-1)=(1.-prelax)*pp(k,M-1,P-1) + prelax*x(k+1)
 1107 continue


c     ********************End of Pressure****************************
      errorn=0.0
      errord=0.0
      do 1325 k=0,P-1,1
      do 1310 i=0,N-1,1
        do 1320 j=0,M-1,1
          errorn=errorn+ abs(pp(i,j,k)-ppold(i,j,k))
          errord=errord+abs(pp(i,j,k))
 1320 continue
 1310 continue
 1325 continue
      error=errorn/errord
c     write(6,*)"pressure error = " ,error
 1400 continue
      ppavg=0.0
      do 1309 k=0,P-1,1
       do 1300 i=0,N-1,1
```

```
        do 1200 j=0,M-1,1
           ppavg=ppavg + abs(pp(i,j,k))
           press(i,j,k)=press(i,j,k)+0.2*pp(i,j,k)
 1200      continue
 1300 continue
 1309 continue
      ppavg=ppavg/(N*M*P)
c      write(6,*) " pp avg= ",ppavg
      avgpress=0.0
      pcorrect=press(2,2,2)
      do 1349 k=0,P-1,1
       do 1349 i=0,N-1,1
         do 1349 j=0,M-1,1
           press(i,j,k)=press(i,j,k)-pcorrect
           avgpress=avgpress  + abs(press(i,j,k))
 1349 continue
      avgpress=avgpress/(N*M*P)
c      write(6,*) "avg pressure= ",avgpress
c     Velocity correction equations
      do 2000 k=0,P-1,1
      do 2000 i=0,N-2,1
        do 2000 j=0,M-1,1
        dYp=ycord(j+1)-ycord(j)
        dZp=zcord(k+1)-Zcord(k)
        u(i+1,j+1,k+1)= u(i+1,j+1,k+1) +
     &        (dYp*dZp*(pp(i,j,k)-pp(i+1,j,k))/(Apu(i,j,k)))*0.2



 2000 continue

      do 2200 k=0,P-1,1
      do 2200 i=0,N-1,1
        do 2200 j=0,M-2,1
        dXp=xcord(i+1)-xcord(i)
        dZp=zcord(k+1)-zcord(k)
        v(i+1,j+1,k+1)= v(i+1,j+1,k+1) +
     &         (dXp*dZp*(pp(i,j,k)-pp(i,j+1,k))/(Apv(i,j,k)))*0.2

 2200 continue


      do 2246 k=0,P-2,1
      do 2246 i=0,N-1,1
        do 2246 j=0,M-1,1
        dXp=xcord(i+1)-xcord(i)
        dYp=ycord(j+1)-ycord(j)
        w(i+1,j+1,k+1)= w(i+1,j+1,k+1) +
     &         (dXp*dYp*(pp(i,j,k)-pp(i,j,k+1))/(Apw(i,j,k)))*0.2


 2246 continue
      return
c     End of Subroutine
      end
```

```
c    subroutine for creating a grid
     Subroutine Grid_form(N,M,P,len,bre,hei)
     Integer N,M,P
     Real len,bre,hei,PI,C

     Common/grid/Xcord(0:40),Ycord(0:40),Zcord(0:40)
          PI = ACOS(-1.)
          Xcord(0)=0.0
          Xcord(N)=len
          C=0.0
          do 9  i=0,N,1
                  C=C + sin(PI*i/(N+1))
  9 continue
          C=len/C
     do 10 i=1,N-1,1
              Xcord(i)=Xcord(i-1) + C*sin(PI*i/(N+1))
  10    continue
          Ycord(0)=0.0
          Ycord(M)=bre
          C=0.0
          do 19  j=0,M,1
                  C=C + sin(PI*j/(M+1))
  19 continue
          C=bre/C
     do 11 j=1,M-1,1
              Ycord(j)=Ycord(j-1) + C*sin(PI*j/(M+1))

  11    continue
          Zcord(0)=0.0
          C=0.0
          do 29  k=0,P,1
                  C=C + sin((PI/2)*k/(P+1))
  29 continue
          C=hei/C
     do 12 k=1,(P-1),1
           Zcord(k)=Zcord(k-1) + C*sin((PI/2)*k/(P+1))
  12    continue
     Zcord(P)=hei

     return
     end
c    End of subroutine for creating the grid

c    Function for interpolating
     Real Function velonew(v1,x1,v2,x2)
      real v1,x1,v2,x2
     velonew=(v1*x2+v2*x1)/(x1+x2)
     Return
c    End of function velonew
     end
c******************************************************************************************
*******
c    Subroutine for calculating the temperature field
c******************************************************************************************
*******
```

```
      Subroutine TempCalc(u,v,w,N,M,P,temp,Pr,dtime,tempo)
      integer N,M,P
      real temp(0:N+1,0:M+1,0:P+1)
      Common/grid/Xcord(0:40),Ycord(0:40),Zcord(0:40)
      Real Aw,Ae,An,As,At,Ab,De,Dn,Ds,Dw,Dt,Db,Ap,Apo
      Real dXp,dYp,dXw,dXe,dYs,dYn,dZt,dZb
      Real Fe,Fw,Fn,Fs,Ft,Fb
      real a,b,c,d,x
      real dtime
      real Trelax
      Common/thomas/ a(0:40),b(0:40),c(0:40),d(0:40),x(1:41)
      real u(0:N,0:M+1,0:P+1)
      real v(0:N+1,0:M,0:P+1)
      real w(0:N+1,0:M+1,0:P)
      real tempo(0:N+1,0:M+1,0:P+1)
      real error,errorn,errord
      real Kc
      Integer Ttime,Ttmax
      real tempold(0:40,0:40,0:40)
      real ktherm
      real Pr
      real dm3
c     Beginning of executable code

      dm3=1.0/Pr
      Kc=0.0
      Trelax=0.8
      error=1.
      Ttime=0
      Ttmax=80
        do 746 i=0,N+1,1
           do 746 j=0,M+1,1
               temp(i,j,0)=1.00
               temp(i,j,P+1)=0.00
  746 continue

      do 1400 while (error.gt.0.0000001.and.Ttime.lt.Ttmax)
      Ttime=Ttime + 1
      do 400 k=0,P+1,1
         do 400 i=0,N+1,1
            do 400 j=0,M+1,1
               tempold(i,j,k)=temp(i,j,k)
  400    continue

c*************************** For the lower most plane ***************************
c     For the first row of elements
c     For the very first cell only
      dXp=Xcord(1)-Xcord(0)
      dYp=Ycord(1)-Ycord(0)
      dZp=Zcord(1)-Zcord(0)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
      dZb=(Zcord(1)-Zcord(0))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,1,1)
      Fw=dYp*dZp*u(0,1,1)
      Fn=dXp*dZp*v(1,1,1)
      Fs=dXp*dZp*v(1,0,1)
      Ft=dXp*dYp*w(1,1,1)
      Fb=dXp*dYp*w(1,1,0)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c     For newmen boundary condition at the west boundary when flux is non-zero
      temp(0,1,1)=temp(1,1,1)-kc*dXw
      temp(1,0,1)=temp(1,1,1)-kc*dYs

      d(0)=Aw*temp(0,1,1)+ An*temp(1,2,1)+ As*temp(1,0,1)+
     &        At*temp(1,1,2) + Ab*temp(1,1,0)+Apo*tempo(1,1,1)
c     For middle cells in the first row
      do 500 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(1)-Zcord(0)
      dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
      dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
      dZb=(Zcord(1)-Zcord(0))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(k+1,1,1)
      Fw=dYp*dZp*u(k,1,1)
      Fn=dXp*dZp*v(k+1,1,1)
      Fs=dXp*dZp*v(k+1,0,1)
      Ft=dXp*dYp*w(k+1,1,1)
      Fb=dXp*dYp*w(k+1,1,0)
      Ae=De*greater(0.,1-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
c     Newmann boundary condition of zero flux
      temp(k+1,0,1)=temp(k+1,1,1)-kc*dYs
      d(k)=An*temp(k+1,2,1) + As*temp(k+1,0,1)+ At*temp(k+1,1,2) +
     &          Ab*temp(k+1,1,0)+Apo*tempo(k+1,1,1)
 500  continue
c     for last cell in the first row
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(1)-Zcord(0)
      dXe=(Xcord(N)-Xcord(N-1))/2
      dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
      dZb=(Zcord(1)-Zcord(0))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(N,1,1)
      Fw=dYp*dZp*u(N-1,1,1)
      Fn=dXp*dZp*v(N,1,1)
      Fs=dXp*dZp*v(N,0,1)
      Ft=dXp*dYp*w(N,1,1)
      Fb=dXp*dYp*w(N,1,0)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(N-1)=-Aw
      b(N-1)=Ap
c     Applying the newmann BC of zero flux
      temp(N+1,1,1)=temp(N,1,1)-Kc*dXe
      temp(N,0,1)=temp(N,1,1)-Kc*dYs
      d(N-1)=An*temp(N,2,1) + As*temp(N,0,1) + Ae*temp(N+1,1,1) +
     &       At*temp(N,1,2) + Ab*temp(N,1,0)+Apo*tempo(N,1,1)
c     call thomas algorithm
      call Thomas_algo(N)
      do 600 k=0,N-1,1
      temp(k+1,1,1)=(1.-trelax)*temp(k+1,1,1) + trelax*x(k+1)
 600  continue
c     for middle row of elements
c     for first element in the row
      do 800 j=1,M-2,1
      dXp=xcord(1)-xcord(0)
      dYp=ycord(j+1)-ycord(j)
      dZp=Zcord(1)-Zcord(0)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
      dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j)+Ycord(j-1))/2
      dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
      dZb=(Zcord(1)-Zcord(0))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,j+1,1)
      Fw=dYp*dZp*u(0,j+1,1)
      Fn=dXp*dZp*v(1,j+1,1)
      Fs=dXp*dZp*v(1,j,1)
      Ft=dXp*dYp*w(1,j+1,1)
      Fb=dXp*dYp*w(1,j+1,0)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c     For newmen boundary condition at the west boundary when flux is zero
      temp(0,1+j,1)=temp(1,1+j,1)-Kc*dXw
      d(0)=Aw*temp(0,j+1,1) + An*temp(1,j+2,1)+ As*temp(1,j,1)+
     &      At*temp(1,j+1,2) + Ab*temp(1,j+1,0) + Apo*tempo(1,j+1,1)
c     For middle elements in the row
      do 700 k=1,N-2,1
        dXp=xcord(k+1)-xcord(k)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(1)-Zcord(0)
        dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
        dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
        dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
        dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
        dZb=(Zcord(1)-Zcord(0))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(k+1,j+1,1)
        Fw=dYp*dZp*u(k,j+1,1)
        Fn=dXp*dZp*v(k+1,j+1,1)
        Fs=dXp*dZp*v(k+1,j,1)
```

```
          Ft=dXp*dYp*w(k+1,j+1,1)
          Fb=dXp*dYp*w(k+1,j+1,0)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
          a(k)=-Aw
          b(k)=Ap
          c(k)=-Ae
          d(k)=An*temp(k+1,j+2,1)+ As*temp(k+1,j,1) +
     &   At*temp(k+1,j+1,2) + Ab*temp(k+1,j+1,0)+ Apo*tempo(k+1,j+1,1)
     &
  700     continue
c    For last element in the row
        dXp=xcord(N)-xcord(N-1)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(1)-Zcord(0)
        dXe=(Xcord(N)-Xcord(N-1))/2
        dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
        dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
        dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
        dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
        dZb=(Zcord(1)-Zcord(0))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(N,j+1,1)
        Fw=dYp*dZp*u(N-1,j+1,1)
        Fn=dXp*dZp*v(N,j+1,1)
        Fs=dXp*dZp*v(N,j,1)
        Ft=dXp*dYp*w(N,j+1,1)
        Fb=dXp*dYp*w(N,j+1,0)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        a(N-1)=-Aw
        b(N-1)=Ap
c    Applying the newmann BC of zero flux
        temp(N+1,1+j,1)=temp(N,1+j,1)-Kc*dXe
        d(N-1)=An*temp(N,j+2,1)+ As*temp(N,j,1) + Ae*temp(N+1,j+1,1) +
     &   At*temp(N,j+1,2) + Ab*temp(N,j+1,0) + Apo*tempo(N,j+1,1)
c    call thomas algorithm
      call Thomas_algo(N)
      do 900 k=0,N-1,1
      temp(k+1,j+1,1)=(1.-trelax)*temp(k+1,j+1,1) + trelax*x(k+1)
  900 continue

  800 continue
c     for last row of elements
c     For first element of the row
      dXp=xcord(1)-xcord(0)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(1)-Zcord(0)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(M)-Ycord(M-1))/2
      dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
      dZb=(Zcord(1)-Zcord(0))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,M,1)
      Fw=dYp*dZp*u(0,M,1)
      Fn=dXp*dZp*v(1,M,1)
      Fs=dXp*dZp*v(1,M-1,1)
      Ft=dXp*dYp*w(1,M,1)
      Fb=dXp*dYp*w(1,M,0)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c     For newmen boundary condition at the west & north boundary when flux is zero
      temp(0,M,1)=temp(1,M,1)-Kc*dXw
      temp(1,M+1,1)=temp(1,M,1)-Kc*dYn
      d(0)=Aw*temp(0,M,1) + An*temp(1,M+1,1) + As*temp(1,M-1,1) +
     &     At*temp(1,M,2) + Ab*temp(1,M,0) + Apo*tempo(1,M,1)
c     For middle row of elements
      do 1000 k=1,N-2,1
        dXp=xcord(k+1)-xcord(k)
        dYp=ycord(M)-ycord(M-1)
        dZp=Zcord(1)-Zcord(0)
        dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
        dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dYn=(Ycord(M)-Ycord(M-1))/2
        dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
        dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
        dZb=(Zcord(1)-Zcord(0))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
```

```
            Ds=dm3*dXp*dZp/dYs
            Dt=dm3*dXp*dYp/dZt
            Db=dm3*dXp*dYp/dZb
            Fe=dYp*dZp*u(k+1,M,1)
            Fw=dYp*dZp*u(k,M,1)
            Fn=dXp*dZp*v(k+1,M,1)
            Fs=dXp*dZp*v(k+1,M-1,1)
            Ft=dXp*dYp*w(k+1,M,1)
            Fb=dXp*dYp*w(k+1,M,0)
            Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
            Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
            An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
            As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
            At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
            Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
            Apo=dXp*dYp*dZp/dtime
            Ap=Ae+Aw+An+As+At+Ab+Apo
            a(k)=-Aw
            b(k)=Ap
            c(k)=-Ae
c    Newmann conditon of zero flux on the north boundary
            temp(k+1,M+1,1)=temp(k+1,M,1)-Kc*dYn
            d(k)=As*temp(k+1,M-1,1) + An*temp(k+1,M+1,1) +
     &         At*temp(k+1,M,2) + Ab*temp(k+1,M,0) + Apo*tempo(k+1,M,1)
 1000 continue
c    For last element of the row
          dXp=xcord(N)-xcord(N-1)
          dYp=ycord(M)-ycord(M-1)
          dZp=Zcord(1)-Zcord(0)
          dXe=(Xcord(N)-Xcord(N-1))/2
          dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
          dYn=(Ycord(M)-Ycord(M-1))/2
          dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
          dZt=(Zcord(2)+Zcord(1))/2-(Zcord(1)+Zcord(0))/2
          dZb=(Zcord(1)-Zcord(0))/2
          De=dm3*dYp*dZp/dXe
          Dw=dm3*dYp*dZp/dXw
          Dn=dm3*dXp*dZp/dYn
          Ds=dm3*dXp*dZp/dYs
          Dt=dm3*dXp*dYp/dZt
          Db=dm3*dXp*dYp/dZb

          Fe=dYp*dZp*u(N,M,1)
          Fw=dYp*dZp*u(N-1,M,1)
          Fn=dXp*dZp*v(N,M,1)
          Fs=dXp*dZp*v(N,M-1,1)
          Ft=dXp*dYp*w(N,M,1)
          Fb=dXp*dYp*w(N,M,0)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
          At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
          Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
          Apo=dXp*dYp*dZp/dtime
          Ap=Ae+Aw+An+As+At+Ab+Apo
```

```
      a(N-1)=-Aw
      b(N-1)=Ap
c     Applying the newmann BC of zero flux at north and east boundaries
      temp(N+1,M,1)=temp(N,M,1)-Kc*dXe
      temp(N,M+1,1)=temp(N,M,1)-Kc*dYn

      d(N-1)=As*temp(N,M-1,1) + An*temp(N,M+1,1) + Ae*temp(N+1,M,1) +
     &     At*temp(N,M,2)+ Ab*temp(N,M,0) + Apo*tempo(N,M,1)
c     call thomas algorithm
      call Thomas_algo(N)
      do 1100 k=0,N-1,1
      temp(k+1,M,1)=(1.-trelax)*temp(k+1,M,1) + trelax*x(k+1)
 1100 continue
c************* End of Lower most plane ****************************************

c************* For the middle planes *****************************************
      do 1 f=1,P-2,1
c     For the first row of elements
c     For the very first cell only
      dXp=Xcord(1)-Xcord(0)
      dYp=Ycord(1)-Ycord(0)
      dZp=Zcord(f+1)-Zcord(f)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,1,f+1)
      Fw=dYp*dZp*u(0,1,f+1)
      Fn=dXp*dZp*v(1,1,f+1)
      Fs=dXp*dZp*v(1,0,f+1)
      Ft=dXp*dYp*w(1,1,f+1)
      Fb=dXp*dYp*w(1,1,f)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c     For newmen boundary condition at the west boundary when flux is zero
      temp(0,1,f+1)=temp(1,1,f+1)-Kc*dXw
      temp(1,0,f+1)=temp(1,1,f+1)-Kc*dYs

      d(0)=Aw*temp(0,1,f+1)+ An*temp(1,2,f+1)+ As*temp(1,0,f+1)+
     &      At*temp(1,1,f+2) + Ab*temp(1,1,f) + Apo*tempo(1,1,f+1)
```

```
c     For middle cells in the first row
      do 501 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(f+1)-Zcord(f)
      dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k+0))/2
      dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(k+1,1,f+1)
      Fw=dYp*dZp*u(k,1,f+1)
      Fn=dXp*dZp*v(k+1,1,f+1)
      Fs=dXp*dZp*v(k+1,0,f+1)
      Ft=dXp*dYp*w(k+1,1,f+1)
      Fb=dXp*dYp*w(k+1,1,f)
      Ae=De*greater(0.,1-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
c     Newmann boundary condition of zero flux
      temp(k+1,0,f+1)=temp(k+1,1,f+1)-Kc*dYs
      d(k)=An*temp(k+1,2,f+1) + As*temp(k+1,0,f+1)+ At*temp(k+1,1,f+2) +
     &          Ab*temp(k+1,1,f)+Apo*tempo(k+1,1,f+1)
 501  continue
c     for last cell in the first row
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(f+1)-Zcord(f)
      dXe=(Xcord(N)-Xcord(N-1))/2
      dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
```

```
       Fe=dYp*dZp*u(N,1,f+1)
       Fw=dYp*dZp*u(N-1,1,f+1)
       Fn=dXp*dZp*v(N,1,f+1)
       Fs=dXp*dZp*v(N,0,f+1)
       Ft=dXp*dYp*w(N,1,f+1)
       Fb=dXp*dYp*w(N,1,f)
       Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
       Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
       An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
       As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
       At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
       Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
       Apo=dXp*dYp*dZp/dtime
       Ap=Ae+Aw+An+As+At+Ab+Apo
       a(N-1)=-Aw
       b(N-1)=Ap
c      Applying the newmann BC of zero flux
      temp(N+1,1,f+1)=temp(N,1,f+1)-Kc*dXe
      temp(N,0,f+1)=temp(N,1,f+1)-Kc*dYs
      d(N-1)=An*temp(N,2,f+1) + As*temp(N,0,f+1) + Ae*temp(N+1,1,f+1) +
     &       At*temp(N,1,f+2) + Ab*temp(N,1,f) + Apo*tempo(N,1,f+1)
c      call thomas algorithm
      call Thomas_algo(N)
      do 601 k=0,N-1,1
      temp(k+1,1,f+1)=(1.-trelax)*temp(k+1,1,f+1) + trelax*x(k+1)
  601 continue
c      for middle row of elements
c      for first element in the row
      do 801 j=1,M-2,1
        dXp=xcord(1)-xcord(0)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(f+1)-Zcord(f)
        dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
        dXw=(Xcord(1)-Xcord(0))/2
        dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
        dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j)+Ycord(j-1))/2
        dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb


        Fe=dYp*dZp*u(1,j+1,f+1)
        Fw=dYp*dZp*u(0,j+1,f+1)
        Fn=dXp*dZp*v(1,j+1,f+1)
        Fs=dXp*dZp*v(1,j,f+1)
        Ft=dXp*dYp*w(1,j+1,f+1)
        Fb=dXp*dYp*w(1,j+1,f)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        b(0)=Ap
        c(0)=-Ae
c     For newmen boundary condition at the west boundary when flux is zero
       temp(0,1+j,f+1)=temp(1,j+1,f+1)-Kc*dXw
        d(0)=Aw*temp(0,j+1,f+1) +An*temp(1,j+2,f+1)+ As*temp(1,j,f+1)+
     &     At*temp(1,j+1,f+2) + Ab*temp(1,j+1,f)+Apo*tempo(1,j+1,f+1)
c     For middle elements in the row
        do 701 k=1,N-2,1
           dXp=xcord(k+1)-xcord(k)
           dYp=ycord(j+1)-ycord(j)
           dZp=Zcord(f+1)-Zcord(f)
           dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
           dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
           dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
           dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
           dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
           dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
           De=dm3*dYp*dZp/dXe
           Dw=dm3*dYp*dZp/dXw
           Dn=dm3*dXp*dZp/dYn
           Ds=dm3*dXp*dZp/dYs
           Dt=dm3*dXp*dYp/dZt
           Db=dm3*dXp*dYp/dZb

           Fe=dYp*dZp*u(k+1,j+1,f+1)
           Fw=dYp*dZp*u(k,j+1,f+1)
           Fn=dXp*dZp*v(k+1,j+1,f+1)
           Fs=dXp*dZp*v(k+1,j,f+1)
           Ft=dXp*dYp*w(k+1,j+1,f+1)
           Fb=dXp*dYp*w(k+1,j+1,f)
           Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
           Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
           An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
           As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
           At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
           Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
           Apo=dXp*dYp*dZp/dtime
           Ap=Ae+Aw+An+As+At+Ab+Apo
           a(k)=-Aw
           b(k)=Ap
           c(k)=-Ae
           d(k)=An*temp(k+1,j+2,f+1)+ As*temp(k+1,j,f+1) +
     &          At*temp(k+1,j+1,f+2) + Ab*temp(k+1,j+1,f)
     &          + Apo*tempo(k+1,j+1,f+1)
701        continue
c     For last element in the row
        dXp=xcord(N)-xcord(N-1)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(f+1)-Zcord(f)
        dXe=(Xcord(N)-Xcord(N-1))/2
        dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
        dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
        dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
        dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(N,j+1,f+1)
        Fw=dYp*dZp*u(N-1,j+1,f+1)
        Fn=dXp*dZp*v(N,j+1,f+1)
        Fs=dXp*dZp*v(N,j,f+1)
        Ft=dXp*dYp*w(N,j+1,f+1)
        Fb=dXp*dYp*w(N,j+1,f)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        a(N-1)=-Aw
        b(N-1)=Ap
c    Applying the newmann BC of zero flux
        temp(N+1,1+j,f+1)=temp(N,j+1,f+1)-Kc*dXe
        d(N-1)=An*temp(N,j+2,f+1)+ As*temp(N,j,f+1) +
   &     Ae*temp(N+1,j+1,f+1)+ At*temp(N,j+1,f+2)+ Ab*temp(N,j+1,f)
   &       + Apo*tempo(N,j+1,f+1)
c    call thomas algorithm
      call Thomas_algo(N)
      do 901 k=0,N-1,1
      temp(k+1,j+1,f+1)=(1.-trelax)*temp(k+1,j+1,f+1) + trelax*x(k+1)
 901 continue

 801 continue
c    for last row of elements
c    For first element of the row
      dXp=xcord(1)-xcord(0)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(f+1)-Zcord(f)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(M)-Ycord(M-1))/2
      dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,M,f+1)
      Fw=dYp*dZp*u(0,M,f+1)
      Fn=dXp*dZp*v(1,M,f+1)
      Fs=dXp*dZp*v(1,M-1,f+1)
      Ft=dXp*dYp*w(1,M,f+1)
      Fb=dXp*dYp*w(1,M,f)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c     For newmen boundary condition at the west & north boundary when flux is zero
      temp(0,M,f+1)=temp(1,M,f+1)-Kc*dXw
      temp(1,M+1,f+1)=temp(1,M,f+1)-Kc*dYn
      d(0)=Aw*temp(0,M,f+1) + An*temp(1,M+1,f+1) + As*temp(1,M-1,f+1) +
     &    At*temp(1,M,f+2) + Ab*temp(1,M,f) + Apo*tempo(1,M,f+1)
c     For middle row of elements
      do 1001 k=1,N-2,1
        dXp=xcord(k+1)-xcord(k)
        dYp=ycord(M)-ycord(M-1)
        dZp=Zcord(f+1)-Zcord(f)
        dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
        dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dYn=(Ycord(M)-Ycord(M-1))/2
        dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
        dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
        dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(k+1,M,f+1)
        Fw=dYp*dZp*u(k,M,f+1)
        Fn=dXp*dZp*v(k+1,M,f+1)
        Fs=dXp*dZp*v(k+1,M-1,f+1)
        Ft=dXp*dYp*w(k+1,M,f+1)
        Fb=dXp*dYp*w(k+1,M,f)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        a(k)=-Aw
        b(k)=Ap
        c(k)=-Ae
c     Newmann conditon of zero flux on the north boundary
        temp(k+1,M+1,f+1)=temp(k+1,M,f+1)-Kc*dYn
        d(k)=As*temp(k+1,M-1,f+1) + An*temp(k+1,M+1,f+1) +
     &     At*temp(k+1,M,f+2) + Ab*temp(k+1,M,f) +Apo*tempo(k+1,M,f+1)

  1001 continue
c    For last element of the row
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(f+1)-Zcord(f)
      dXe=(Xcord(N)-Xcord(N-1))/2
      dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
      dYn=(Ycord(M)-Ycord(M-1))/2
      dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dZt=(Zcord(f+2)+Zcord(f+1))/2-(Zcord(f+1)+Zcord(f))/2
      dZb=(Zcord(f+1)+Zcord(f))/2-(Zcord(f)+Zcord(f-1))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb

      Fe=dYp*dZp*u(N,M,f+1)
      Fw=dYp*dZp*u(N-1,M,f+1)
      Fn=dXp*dZp*v(N,M,f+1)
      Fs=dXp*dZp*v(N,M-1,f+1)
      Ft=dXp*dYp*w(N,M,f+1)
      Fb=dXp*dYp*w(N,M,f)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(N-1)=-Aw
      b(N-1)=Ap
c    Applying the newmann BC of zero flux at north and east boundaries
     temp(N+1,M,f+1)=temp(N,M,f+1)-Kc*dXe
     temp(N,M+1,f+1)=temp(N,M,f+1)-Kc*dYn

      d(N-1)=As*temp(N,M-1,f+1) + An*temp(N,M+1,f+1) +
     &    Ae*temp(N+1,M,f+1)+ At*temp(N,M,f+2)+ Ab*temp(N,M,f)
     &    + Apo*tempo(N,M,f+1)
c    call thomas algorithm
     call Thomas_algo(N)
     do 1101 k=0,N-1,1
     temp(k+1,M,f+1)=(1.-trelax)*temp(k+1,M,f+1) + trelax*x(k+1)
 1101 continue


   1 Continue
c************* End of middle planes ******************************************

c************* For top most plane ********************************************

c    For the first row of elements
```

```
c    For the very first cell only
      dXp=Xcord(1)-Xcord(0)
      dYp=Ycord(1)-Ycord(0)
      dZp=Zcord(P)-Zcord(P-1)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(P)-Zcord(P-1))/2
      dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,1,P)
      Fw=dYp*dZp*u(0,1,P)
      Fn=dXp*dZp*v(1,1,P)
      Fs=dXp*dZp*v(1,0,P)
      Ft=dXp*dYp*w(1,1,P)
      Fb=dXp*dYp*w(1,1,P-1)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c    For newmen boundary condition at the west boundary when flux is zero
      temp(0,1,P)=temp(1,1,P)-Kc*dXw
      temp(1,0,P)=temp(1,1,P)-Kc*dYs

      d(0)=Aw*temp(0,1,P)+ An*temp(1,2,P)+ As*temp(1,0,P)+
     &       At*temp(1,1,P+1) + Ab*temp(1,1,P-1) + Apo*tempo(1,1,P)
c    For middle cells in the first row
      do 502 k=1,N-2,1
      dXp=xcord(k+1)-xcord(k)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(P)-Zcord(P-1)
      dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
      dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(P)-Zcord(P-1))/2
      dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(k+1,1,P)
      Fw=dYp*dZp*u(k,1,P)
      Fn=dXp*dZp*v(k+1,1,P)
      Fs=dXp*dZp*v(k+1,0,P)
      Ft=dXp*dYp*w(k+1,1,P)
      Fb=dXp*dYp*w(k+1,1,P-1)
      Ae=De*greater(0.,1-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(k)=-Aw
      b(k)=Ap
      c(k)=-Ae
c     Newmann boundary condition of zero flux
      temp(k+1,0,P)=temp(k+1,1,P)-Kc*dYs
      d(k)=An*temp(k+1,2,P) + As*temp(k+1,0,P)+ At*temp(k+1,1,P+1) +
     &          Ab*temp(k+1,1,P-1)  + Apo*tempo(k+1,1,P)
  502 continue
c     for last cell in the first row
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(1)-ycord(0)
      dZp=Zcord(P)-Zcord(P-1)
      dXe=(Xcord(N)-Xcord(N-1))/2
      dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
      dYn=(Ycord(2)+Ycord(1))/2-(Ycord(1)+Ycord(0))/2
      dYs=(Ycord(1)-Ycord(0))/2
      dZt=(Zcord(P)-Zcord(P-1))/2
      dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb

      Fe=dYp*dZp*u(N,1,P)
      Fw=dYp*dZp*u(N-1,1,P)
      Fn=dXp*dZp*v(N,1,P)
      Fs=dXp*dZp*v(N,0,P)
      Ft=dXp*dYp*w(N,1,P)
      Fb=dXp*dYp*w(N,1,P-1)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(N-1)=-Aw
      b(N-1)=Ap
c     Applying the newmann BC of zero flux
      temp(N+1,1,P)=temp(N,1,P)-Kc*dXe
      temp(N,0,P)=temp(N,1,P)-Kc*dYs
      d(N-1)=An*temp(N,2,P) + As*temp(N,0,P) + Ae*temp(N+1,1,P) +
     &        At*temp(N,1,P+1) + Ab*temp(N,1,P-1) +Apo*tempo(N,1,P)
c     call thomas algorithm
      call Thomas_algo(N)
      do 602 k=0,N-1,1
      temp(k+1,1,P)=(1.-trelax)*temp(k+1,1,P) + trelax*x(k+1)
  602 continue
c     for middle row of elements
c     for first element in the row
      do 802 j=1,M-2,1
        dXp=xcord(1)-xcord(0)
        dYp=ycord(j+1)-ycord(j)
        dZp=Zcord(P)-Zcord(P-1)
        dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
        dXw=(Xcord(1)-Xcord(0))/2
        dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
        dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
        dZt=(Zcord(P)-Zcord(P-1))/2
        dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(1,j+1,P)
        Fw=dYp*dZp*u(0,j+1,P)
        Fn=dXp*dZp*v(1,j+1,P)
        Fs=dXp*dZp*v(1,j,P)
        Ft=dXp*dYp*w(1,j+1,P)
        Fb=dXp*dYp*w(1,j+1,P-1)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        b(0)=Ap
        c(0)=-Ae
c     For newmen boundary condition at the west boundary when flux is zero
        temp(0,1+j,P)=temp(1,1+j,P)-Kc*dXw
        d(0)=Aw*temp(0,j+1,P) +An*temp(1,j+2,P)+ As*temp(1,j,P)+
     &     At*temp(1,j+1,P+1) + Ab*temp(1,j+1,P-1)+ Apo*tempo(1,j+1,P)
c     For middle elements in the row
        do 702 k=1,N-2,1
          dXp=xcord(k+1)-xcord(k)
          dYp=ycord(j+1)-ycord(j)
          dZp=Zcord(P)-Zcord(P-1)
          dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
          dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
          dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
          dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
          dZt=(Zcord(P)-Zcord(P-1))/2
```

```
            dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
            De=dm3*dYp*dZp/dXe
            Dw=dm3*dYp*dZp/dXw
            Dn=dm3*dXp*dZp/dYn
            Ds=dm3*dXp*dZp/dYs
            Dt=dm3*dXp*dYp/dZt
            Db=dm3*dXp*dYp/dZb

            Fe=dYp*dZp*u(k+1,j+1,P)
            Fw=dYp*dZp*u(k,j+1,P)
            Fn=dXp*dZp*v(k+1,j+1,P)
            Fs=dXp*dZp*v(k+1,j,P)
            Ft=dXp*dYp*w(k+1,j+1,P)
            Fb=dXp*dYp*w(k+1,j+1,P-1)
            Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
            Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
            An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
            As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
            At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
            Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
            Apo=dXp*dYp*dZp/dtime
            Ap=Ae+Aw+An+As+At+Ab+Apo
            a(k)=-Aw
            b(k)=Ap
            c(k)=-Ae
            d(k)=An*temp(k+1,j+2,P)+ As*temp(k+1,j,P) +
     &         At*temp(k+1,j+1,P+1) + Ab*temp(k+1,j+1,P-1)
     &       + Apo*tempo(k+1,j+1,P)

 702        continue
c     For last element in the row
          dXp=xcord(N)-xcord(N-1)
          dYp=ycord(j+1)-ycord(j)
          dZp=Zcord(P)-Zcord(P-1)
          dXe=(Xcord(N)-Xcord(N-1))/2
          dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
          dYn=(Ycord(2+j)+Ycord(1+j))/2-(Ycord(1+j)+Ycord(j))/2
          dYs=(Ycord(1+j)+Ycord(j))/2-(Ycord(j) + Ycord(j-1))/2
          dZt=(Zcord(P)-Zcord(P-1))/2
          dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
          De=dm3*dYp*dZp/dXe
          Dw=dm3*dYp*dZp/dXw
          Dn=dm3*dXp*dZp/dYn
          Ds=dm3*dXp*dZp/dYs
          Dt=dm3*dXp*dYp/dZt
          Db=dm3*dXp*dYp/dZb
          Fe=dYp*dZp*u(N,j+1,P)
          Fw=dYp*dZp*u(N-1,j+1,P)
          Fn=dXp*dZp*v(N,j+1,P)
          Fs=dXp*dZp*v(N,j,P)
          Ft=dXp*dYp*w(N,j+1,P)
          Fb=dXp*dYp*w(N,j+1,P-1)
          Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
          Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
          An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
          As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
```

```
       At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
       Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
       Apo=dXp*dYp*dZp/dtime
       Ap=Ae+Aw+An+As+At+Ab+Apo
       a(N-1)=-Aw
       b(N-1)=Ap
c    Applying the newmann BC of zero flux
       temp(N+1,1+j,P)=temp(N,1+j,P)-Kc*dXe
       d(N-1)=An*temp(N,j+2,P)+ As*temp(N,j,P) +
     &    Ae*temp(N+1,j+1,P)+ At*temp(N,j+1,P+1)+ Ab*temp(N,j+1,P-1)
     &     + Apo*tempo(N,j+1,P)
c    call thomas algorithm
      call Thomas_algo(N)
      do 902 k=0,N-1,1
      temp(k+1,j+1,P)=(1.-trelax)*temp(k+1,j+1,P) + trelax*x(k+1)
 902 continue

 802 continue
c    for last row of elements
c    For first element of the row
      dXp=xcord(1)-xcord(0)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(P)-Zcord(P-1)
      dXe=(Xcord(2)+Xcord(1))/2-(Xcord(1)+Xcord(0))/2
      dXw=(Xcord(1)-Xcord(0))/2
      dYn=(Ycord(M)-Ycord(M-1))/2
      dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dZt=(Zcord(P)-Zcord(P-1))/2
      dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(1,M,P)
      Fw=dYp*dZp*u(0,M,P)
      Fn=dXp*dZp*v(1,M,P)
      Fs=dXp*dZp*v(1,M-1,P)
      Ft=dXp*dYp*w(1,M,P)
      Fb=dXp*dYp*w(1,M,P-1)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      b(0)=Ap
      c(0)=-Ae
c    For newmen boundary condition at the west & north boundary when flux is zero
     temp(0,M,P)=temp(1,M,P)-Kc*dXw
     temp(1,M+1,P)=temp(1,M,P)-Kc*dYn
     d(0)=Aw*temp(0,M,P) + An*temp(1,M+1,P) + As*temp(1,M-1,P) +
    &    At*temp(1,M,P+1) + Ab*temp(1,M,P-1)+ Apo*tempo(1,M,P)
```

```
c     For middle row of elements
      do 1002 k=1,N-2,1
        dXp=xcord(k+1)-xcord(k)
        dYp=ycord(M)-ycord(M-1)
        dZp=Zcord(P)-Zcord(P-1)
        dXe=(Xcord(k+2)+Xcord(k+1))/2-(Xcord(k+1)+Xcord(k))/2
        dXw=(Xcord(k+1)+Xcord(k))/2-(Xcord(k)+Xcord(k-1))/2
        dYn=(Ycord(M)-Ycord(M-1))/2
        dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
        dZt=(Zcord(P)-Zcord(P-1))/2
        dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
        De=dm3*dYp*dZp/dXe
        Dw=dm3*dYp*dZp/dXw
        Dn=dm3*dXp*dZp/dYn
        Ds=dm3*dXp*dZp/dYs
        Dt=dm3*dXp*dYp/dZt
        Db=dm3*dXp*dYp/dZb
        Fe=dYp*dZp*u(k+1,M,P)
        Fw=dYp*dZp*u(k,M,P)
        Fn=dXp*dZp*v(k+1,M,P)
        Fs=dXp*dZp*v(k+1,M-1,P)
        Ft=dXp*dYp*w(k+1,M,P)
        Fb=dXp*dYp*w(k+1,M,P-1)
        Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
        Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
        An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
        As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
        At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
        Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
        Apo=dXp*dYp*dZp/dtime
        Ap=Ae+Aw+An+As+At+Ab+Apo
        a(k)=-Aw
        b(k)=Ap
        c(k)=-Ae
c     Newmann conditon of zero flux on the north boundary
        temp(k+1,M+1,P)=temp(k+1,M,P)-Kc*dYn
        d(k)=As*temp(k+1,M-1,P) + An*temp(k+1,M+1,P) +
     &    At*temp(k+1,M,P+1) + Ab*temp(k+1,M,P-1) + Apo*tempo(k+1,M,P)

 1002 continue
c     For last element of the row
      dXp=xcord(N)-xcord(N-1)
      dYp=ycord(M)-ycord(M-1)
      dZp=Zcord(P)-Zcord(P-1)
      dXe=(Xcord(N)-Xcord(N-1))/2
      dXw=(Xcord(N)+Xcord(N-1))/2-(Xcord(N-1)+Xcord(N-2))/2
      dYn=(Ycord(M)-Ycord(M-1))/2
      dYs=(Ycord(M)+Ycord(M-1))/2-(Ycord(M-1)+Ycord(M-2))/2
      dZt=(Zcord(P)-Zcord(P-1))/2
      dZb=(Zcord(P)+Zcord(P-1))/2-(Zcord(P-1)+Zcord(P-2))/2
      De=dm3*dYp*dZp/dXe
      Dw=dm3*dYp*dZp/dXw
      Dn=dm3*dXp*dZp/dYn
      Ds=dm3*dXp*dZp/dYs
      Dt=dm3*dXp*dYp/dZt
      Db=dm3*dXp*dYp/dZb
      Fe=dYp*dZp*u(N,M,P)
      Fw=dYp*dZp*u(N-1,M,P)
      Fn=dXp*dZp*v(N,M,P)
      Fs=dXp*dZp*v(N,M-1,P)
      Ft=dXp*dYp*w(N,M,P)
      Fb=dXp*dYp*w(N,M,P-1)
      Ae=De*greater(0.,1.-0.5*abs(Fe/De)) + greater(-Fe,0.)
      Aw=Dw*greater(0.,1.-0.5*abs(Fw/Dw)) + greater(Fw,0.)
      An=Dn*greater(0.,1.-0.5*abs(Fn/Dn)) + greater(-Fn,0.)
      As=Ds*greater(0.,1.-0.5*abs(Fs/Ds)) + greater(Fs,0.)
      At=Dt*greater(0.,1.-0.5*abs(Ft/Dt)) + greater(-Ft,0.)
      Ab=Db*greater(0.,1.-0.5*abs(Fb/Db)) + greater(Fb,0.)
      Apo=dXp*dYp*dZp/dtime
      Ap=Ae+Aw+An+As+At+Ab+Apo
      a(N-1)=-Aw
      b(N-1)=Ap
c     Applying the newmann BC of zero flux at north and east boundaries
      temp(N+1,M,P)=temp(N,M,P)-Kc*dXe
      temp(N,M+1,P)=temp(N,M,P)-Kc*dYn

      d(N-1)=As*temp(N,M-1,P) + An*temp(N,M+1,P) + Ae*temp(N+1,M,P) +
     &      At*temp(N,M,P+1)+ Ab*temp(N,M,P-1)+ Apo*tempo(N,M,P)
c     call thomas algorithm
      call Thomas_algo(N)
      do 1102 k=0,N-1,1
      temp(k+1,M,P)=(1.-trelax)*temp(k+1,M,P) + trelax*x(k+1)
 1102 continue



c************* End of top most plane ******************************************
      errorn=0.0
      errord=0.0
      do 1310 k=0,P+1,1
         do 1310 i=0,N+1,1
            do 1310 j=0,M+1,1
               errorn=errorn+ abs(temp(i,j,k)-tempold(i,j,k))
               errord=errord+ abs(temp(i,j,k))
 1310     continue

      error=errorn/errord
c      write(6,*)"temperature error = " ,error
 1400 continue
      return
      end
c     End of subroutine TempCalc

c     Function for calculating the greater of two values
      Real Function greater(a,b)
      real a,b
      if (a.ge.b) then
                  greater=a
            else
                  greater=b
         End If
      Return
c     End of Function greater
      end


c     Sub program for Thomas Algorithm
      Subroutine Thomas_algo(number)
c     Variable Declaration
      integer number
      Real k(1:60),l(1:60)
      Real a,b,c,d,x
      common/thomas/a(0:40),b(0:40),c(0:40),d(0:40),x(1:41)
      a(0)=0
      c(number-1)=0
      k(1)=d(0)/b(0)
      l(1)=c(0)/b(0)
      do 300 i=2,number,1
         k(i)=(d(i-1)-a(i-1)*k(i-1))/(b(i-1)-a(i-1)*l(i-1))
         l(i)=(c(i-1))/(b(i-1)-a(i-1)*l(i-1))
  300 continue
      x(number)=k(number)
      do 400 i=number-1,1,-1
         x(i)=k(i)-l(i)*x(i+1)
  400 continue
      return
c     End of sub program Thomas_algo
      end
```

REFERENCES

[Alc64] C. B. Alcock and T. N. Bedford (1964). "Thermodynamics and solubility of oxygen in liquid metals from E.M.F. measurements involving solid electrolytes." *Transactions of the Faraday Society* **60** 822-835.

[Alc77] C. B. Alcock and K. T. Jacob (1977). "Solubility and activity of oxygen in liquid gallium and gallium-copper alloys." *Journal of the Less-Common Metals* **53** 211-222.

[Ane96] L. Anendaño-López, F. L. Castillo-Alvarado, A. Escamilla-Esquivel, G. Contreras-Puente, J. Ortiz-López and O. Zelaya-Angel (1996). "Optical phonons in $Zn_xCd_{1-x}Se$ thin films." *Solid State Communications* **100** 33-36.

[Ant84] T. C. Anthony, A. L. Fahrenbruch and R. H. Bube (1984). "Growth of CdTe films by close-space vapor transport." *Journal of Vacuum Science and Technology A* **2** 1296.

[Apa97] R. Aparicio (1997) Chemical vapor processing of ceramic coatings and composites. Ph.D. Dissertation, University of Florida.

[Bar95] I. Barin (1995). Thermochemical data of pure substances. New York, VCH.

[Bén01] M. H. Bénard (1901). "Les tourbillions cellulaires dans une nappe liquide transportant de la chaleur par convection en régime permanent." *Annales de Chemie et de Physique* **23** 62.

[Ben01] K. Benhadji and P. Vasseur (2001). "Double diffusive convection in a shallow porous cavity filled with a non-Newtonian fluid" *International Communications in Heat and Mass Transfer* **28** 763-772.

[Bir60] R. B. Bird, W. E. Stewart and E. N. Lightfoot (1960). Transport Phenomena. New York, Wiley.

[Bir02] R. B. Bird, W. E. Stewart and E. N. Lightfoot (2002). Transport Phenomena, 2nd Ed. New York, Wiley.

[Bou03] J. Boussinesq (1903). Theorie Analytique de la Chaleur. Paris, Gauthier-Villars.

[Bri25] P. W. Bridgman (1925). *Proceedings of the American Academy of Arts and Sciences* **60** 305.

[Bur93] L.C. Burmeister (1993). Convective Heat Transfer, 2nd Ed. New York, Wiley.

[Bus72] F. H. Busse (1972). "The oscillatory instability of convection rolls in a low Prandtl-number fluid." *Journal of Fluid Mechanics* **52, part 1** 97-112.

[Cat72] I. Catton (1972). "The effect of insulating vertical walls on the onset of motion in a fluid heated from below." *International Journal of Heat and Mass Transfer* **15** 665-672.

[Cat70] I. Catton and E. K. Edwards (1970). "Initiation of thermal convection in finite right circular cylinders." *AIChE Journal* **16** 594.

[Cha61] S. Chandrasekhar (1961). Hydrodynamic and Hydromagnetic Stability. New York, Dover.

[Cha88] S. C. Chapra and R. P. Canale (1988). Numerical Methods for Engineers. New York, McGraw-Hill.

[Cha70] G. S. Charlson and R. L. Sani (1970). "Thermoconvective instability in a bounded cylindrical fluid layer." *International Journal of Heat and Mass Transfer* **13** 1479-1496.

[Cha71] G. S. Charlson and R. L. Sani (1971). "On thermoconvective instability in a bounded cylindrical fluid layer." *International Journal of Heat and Mass Transfer* **14** 2157-2160.

[Cha01] M. Chang and F. Chen (2001). "Flows induced by inclined rotation in directional solidification of variable-viscosity solutions." *Journal of Crystal Growth* **233** 881-896.

[Chh95] R. P. Chhabra (1995). "A simple method for estimating the viscosity of molten metallic alloys." *Journal of Alloys and Compounds* **221** L1-L3.

[Chh93] R. P. Chhabra and A. Tripathi (1993). "A correlation for the viscosity of liquid metals." *High Temperature - High Pressure* **25** 713-718.

[Coh59] M. H. Cohen and D. Turnbull (1959). "Molecular transport in liquids and glasses." *Journal of Chemical Physics* **31** 1164-1169.

[Col00] J. Colombani and J. Bert (2000). "Microgravity and earth diffusion in liquids holographic visualization of convection. " *Space Technologies and Applications International Forum*, ed. M. El-Genk, American Institute of Physics.

[Cot86] D. Côté, J. P. Dodelet, B. A. Lombos and J. I. Dickson (1986). "Epitaxy of GaAs by the Close-Spaced Vapor Transport technique." *Journal of the Electrochemical Society* **133** 1925.

[Cru99] D. W. Crunkleton, N. Gupta, R. Narayanan and T. J. Anderson (1999). "Natural convection of low Prandtl-number fluids in rectangular enclosures." *AIAA Technical Paper 99-0840.*

[Czo18] J. Czockralski (1918). "Ein neunes Vefahren zur Messung der Kristallisationsgeschwindigkeit der Metalle." *Zeitschrift für Physicalische Chemie* **92** 219.

[Das72] S. K. Das and A. Ghosh (1972). "Thermodynamic Measurements in Molten Pb-Sn Alloys." *Metallurgical Transactions* **3** 803-806.

[Der97] R. Derebail and J. N. Koster (1997). "Numerical simulations of natural convection of gallium in a narrow gap." *International Journal of Heat and Mass Transfer* **40** 1169.

[Eis98a] K. Eisele, Z. Zhang, F. Hirt and A. Öngören (1998). "A better view of flows." *Sulzer Technical Review* **1** 14-19.

[Eis98b] K. Eisele, Z. Zhang, F. Hirt and A. Ongoren (1998). "A better view of flows." *Chemical Engineering World* **XXXIII** 47-51.

[Eln71] M. M. A. El-Naggar and N. A. D. Parlee (1971). "Diffusion studies of oxygen in liquid copper and copper alloys by a solid state electrolyte cell technique." *High Temperature Science* **3** 138-154.

[Fah83] R. W. Fahen (1983). Fundamentals of Transport Phenomena. New York, McGraw-Hill.

[Gri92] M. Griglione and T. Anderson (1992). *AIAA Technical Paper*.

[Gup98] N. Gupta (1998) Natural convective flow of low Prandtl number fluids confined in a rectangular box heated from below. M.S. Thesis, University of Florida.

[Gur89] L. V. Gurvich and I. V. Veyts, Eds. (1989). Thermodynamic properties of individual substances. New York, Hemisphere.

[Hah77] S. K. Hahn and D. A. Stevenson (1977). "Oxygen diffusion in the Ga-In-O system." *High Temperature Science* **9** 165-187.

[Hes82a] B. Heshmatpour and D. A. Stevenson (1982). "The influence of solutes on kinetics and thermodynamics of liquid indium-oxygen systems." *Metallurgical Transactions B* **13B** 53-59.

[Hes82b] B. Heshmatpour and D. A. Stevenson (1982). "The influence of solute addition of copper and silver on the diffusivity and thermodynamic properties of oxygen in liquid indium." *Journal of the Electrochemical Society* **129** 430-436.

[Hir54] J. O. Hirschfelder, C. F. Curtiss and R. B. Bird (1954). Molecular Theory of Gases and Liquids. New York, Wiley.

[Hir93] M. Hirai (1993). "Estimation of viscosities of liquid alloys." *ISIJ Journal* **33** 251-258.

[Hon71] S. Honma, N. Sano and Y. Matsushita (1971). "Electrochemical measurements of the diffusivity of oxygen in liquid lead." *Metallurgical Transactions* **2** 1494-1496.

[How97] L. Howle, R. P. Behringer and J. Georgiadis (1997). "Visualization of convective fluid flow in a porous medium." *Nature* **362** 230.

[Hsi97] S.-S. Hsieh and S.-S. Yang (1997). "Flow structure and temperature measurements in a 3-D vertical free convective enclosure at high Rayleigh numbers." *International Journal of Heat and Mass Transfer* **40** 1467-1480.

[Ito88] R. Ito, Y. Inoue, S. Saeki, T. Hirota and K. Nagatomi (1988). "Experimental study of Bénard convection in a square vessel." *Heat Transfer -- Japanese Research* **17** 2.

[Kak95] K. Kakimoto (1995). "Flow Instability during crystal growth from the melt." *Progress in Crystal Growth and Characterization of Materials* **30** 191-215.

[Kak88] K. Kakimoto, M. Eguchi, H. Watanabe and T. Hibiya (1988). "Direct observation by x-ray radiography of convection of molten silicon in the Czochralski growth method." *Journal of Crystal Growth* **88** 365-370.

[Kle82] M. Kleitz, E. Frenandez, J. Fouletier and F. Fabry (1982). Advances in Ceramics, vol. 3. Columbus, OH, American Ceramics Society.

[Kos97] J. N. Koster (1997). "Visualization of Rayleigh-Bénard convection in liquid metals." *European Journal of Mechanics B - Fluids* **16**(3) 447-454.

[Kou96] S. Kou (1996). Transport Phenomena and Materials Processing. New York, Wiley.

[Mar88] B. Martinet and R. J. Adrian (1988). "Rayleigh-Bénard convection: experimental study of time-dependent instabilities." *Experiments in Fluids* **6** 316-322.

[Mat97] B. P. Matisak, A. X. Zhao, R. Narayanan and A. L. Fripp (1997). "The microgravity environment: its prediction, measurement, and importance to materials processing." *Journal of Crystal Growth* **174** 90-95.

[Mat64] Y. Matsushita and K. Goto (1964). "On the application of the oxygen concentration cells with the solid electrolyte ZrO_2-CaO to the basic research works in process metallurgy." *Journal of the Faculty of Engineering, University of Tokyo* **XXVII** 217-280.

[May65] J. E. May (1965). "Kinetics of epitaxial silicon deposition by a low pressure iodide process." *Journal of the Electrochemical Society* **112** 710.

[Mis99] D. Mishra, K. Muralidhar and P. Munshi (1999). "Isotherms in a horizontal differentially heated cavity at intermediate Rayleigh numbers." *International Communications in Heat and Mass Transfer* **26** 729-738.

[Mor90] J. Moreno, J. Jimenez, A. Córdoga, E. Rojas and M. Zamora (1990). "New experimental apparatus for the study of the Rayleigh-Bénard problem." *Review of Scientific Instruments* **51** 82.

[Mul88] G. Müller (1988). Crystal Growths from Melts. New York, Springer-Verlag.

[Mul84] G. Müller, G. Neumann and W. Weber (1984). "Natural convection in vertical Bridgman configurations." *Journal of Crystal Growth* **70** 78-93.

[Nac67] N. H. Nachtrieb (1967). "Self-diffusion in liquid metals." *Advances in Physics* **16** 309-323.

[Nak98] A. Nakano, H. Ozoe and S. W. Churchill (1998). "Numerical computation of natural convection for a low Prandtl-number fluid in a shallow rectangular region heated from below." *Chemical Engineering Journal* **71** 175-182.

[Neu72] P. D. Neufeld, A. R. Janzen and R. A. Aziz (1972). "Empirical equations to calculate 16 of the transport collision integrals $\Omega^{(l,s)*}$ for the Lennard-Jones potential." *Journal of Chemical Physics* **57** 1100.

[Neu90] G. Neumann (1990). "Three-dimensional numerical simulation of buoyancy-driven convection in vertical cylinders heated from below." *Journal of Fluid Mechanics* **214** 559-578.

[Nic63] F. H. Nicoll (1963). "The use of close spacing in chemical-transport systems for growing epitaxial layers of semiconductors." *Journal of the Electrochemical Society* **110** 1165.

[Nie73] R. C. Nielsen and R. H. Sabersky (1973). "Transient heat transfer in Bénard convection." *International Journal of Heat and Mass Transfer* **16** 2407-2420.

[Ola63] D.R. Olander (1963). "Mutual diffusion in dilute solutions." *AIChE Journal* **9** 207.

[Ome96] Omega Complete Temperature Measurement Handbook and Encyclopedia, vol. 29.

[Obe73] K. E. Oberg, L. M. Friedman, W. M. Boorstein and R. A. Rapp (1973). "The diffusivity and solubility of oxygen in liquid copper and liquid silver from electrochemical measurements." *Metallurgical Transactions* **4** 61-67.

[One83] P. V. O'Neil (1983). Advanced Engineering Mathematics, 3rd Ed. Belmont, CA, Wadsworth.

[Ots75] S. Otsuka and Z. Kozuka (1975). "The diffusivity of oxygen in liquid lead by electrochemcial measurements." *Metallurgical Transactions B* **6B** 389-394.

[Ots76] S. Otsuka and Z. Kozuka (1976). "The diffusivity of oxygen in liquid copper by electrochemical measurements." *Metallurgical Transactions B* **7B** 147-149.

[Ots77] S. Otsuka and Z. Kozuka (1977). "The diffusivities of oxygen in nickel and liquid iron determined by electrochemical measurements." *Transactions of the Japan Institute of Metals* **18** 690-696.

[Ots84] S. Otsuka, Z. Kozuka and Y. A. Chang (1984). "Oxygen solubility in liquid indium and oxygen diffusivity in liquid indium and tin." *Metallurgical Transactions B* **15B** 329.

[Ozo76] H. Ozoe, K. Yamamoto, S. W. Churchill and H. Sayama (1976). "Three-dimensional, numerical analysis of laminar convection in a confined fluid heated from below." *Journal of Heat Transfer* **98 Ser C** 202-207.

[Ozo95] H. Ozoe and T. Hara (1995). "Numerical analysis for oscillatory natural convection of low Prandtl number fluid heated from below." *Numerical Heat Transfer, Part A* **27** 307-317.

[Pat80] S. V. Patankar (1980). Numerical Heat Transfer and Fluid Flow. Washington, Hemisphere.

[Per69] L. R. Perkins and C. J. Geankoplis (1969). "Molecular diffusion in a ternary liquid system with the diffusing component dilute." *Chemical Engineering Science* **24** 1035-1042.

[Pra99] S. R. Prasad, C. Malika, T. J. Anderson and R. Narayanan (1999). "An electrochemical method to detect flow profiles during convection in liquid metals." *Journal of Crystal Growth* **198/199** 194-200.

[Pre92] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery (1992). Numerical Recipes in FORTRAN, New York, Cambridge UP.

[Ram72] T. Ramanarayanan and R. Rapp (1972). "The diffusivity and solubility of oxygen in liquid tin and solid silver and the diffusivity of oxygen in solid nickel." *Metallurgical Transactions* **3** 3239-3246.

[Ray16] L. Rayleigh (1916). "On convective currents in a horizontal layer of fluid when the higher temperature is on the underside." *Philosophical Magazine* **32** 529.

[Ros82] S. Rosenblat (1982). "Thermal convection in a vertical cylinder." *Journal of Fluid Mechanics* **122** 395-410.

[Roy92] B. N. Roy (1992). Crystal Growth from Melts. New York, Wiley.

[Sea90] B. R. Sears (1990) Electrochemical Measurements for the Determination of Dynamic States in the Bridgman Crystal Growth Configuration. Ph.D. Dissertation, University of Florida.

[Sea93] B. Sears, T. J. Anderson, R. Narayanan and A. L. Fripp (1993). "The detection of solutal convection during electrochemical measurement of the oxygen diffusivity in liquid tin." *Metallurgical Transactions B* **24B** 91-100.

[Sea92a] B. Sears, R. Narayanan, T. J. Anderson and A. L. Fripp (1992). "Convection of tin in a Bridgman system I. Flow characterization by effective diffusivity measurements." *Journal of Crystal Growth* **125** 404-414.

[Sea92b] B. Sears, A. L. Fripp, J. W.J. Debnam, G. A. Woodell, T. J. Anderson and R. Narayanan (1992). "Convection of tin in a Bridgman system II. An electrochemical method for detecting flow regimes." *Journal of Crystal Growth* **125** 415-419.

[Sha92] J. F. Shackelford (1992). Introduction to Materials Science for Engineers, 3rd Ed. New York, Macmillan.

[Ske74] A. H. P. Skellard (1974). Diffusional Mass Transfer. Malabar, FL, Krieger.

[Smi67] Smithells, C.J., Ed. (1970) Liquid Metals Reference Book. New York, Plenum.

[Ste99] F. Stefani and G. Gerbeth (1999). "Velocity reconstruction in conducting fluids from magnetic field and electric potential measurements." *Inverse Problems* **15** 771-778.

[Swa59] R. A. Swalin (1959). "On the theory of self-diffusion in liquid metals." *Acta Metallurgica* 7 736-740.

[Sze81] S. M. Sze (1981). Physics of Semiconductor Devices. New York, Wiley.

[Szw72] R. Szwarc, K. E. Oberg and R. A. Rapp (1972). "The diffusivity and solubility of oxygen in liquid lead from electrochemical measurements." *High Temperature Science* **4** 347-356.

[Tra69] R. F. Tramposch (1969). "Epitaxial films of germanium deposited on sapphire via chemical vapor transport." *Journal of the Electrochemical Society* **116** 654.

[Tri00] E. Tric, G. Labrosse and M. Betrouni (2000). "A first incursion into the 3D structure of natural convection of air in a differentially heated cubic cavity, from accurate numerical solutions." *International Journal of Heat and Mass Transfer* **43** 4043-4056.

[Tya91] M. S. Tyagi (1991). Introduction to Semiconductors Materials and Devices. New York, Wiley.

[Ued82] M. Ueda, K. Kagawa, K. Yamada, C. Yamaguchi and Y. Harada (1982). "Flow visualization of Bénard convection using holographic inferometry." *Applied Optics* **21** 3269-3272.

[Ups83] C. D. Upson, P. M. Gresho, R. L. Sani, S. T. Chan and R. L. Lee (1983). A thermal convection simulation in three dimensions by a modified finite element method. In Numerical Properties and Methodologies in Heat Transfer, Proceedings of the Second National Symposium, Hemisphere.

[Win85] A. J. A. Winnubst, A. H. A. Scharenborg and A. J. Burggraaf (1985). "Response behaviour of oxygen sensing solid electrolytes." *Journal of Applied Electrochemistry* **15** 139-144.

[Zul00] W. Zulehner (2000). "Historical overview of silicon crystal pulling development." *Materials Science and Engineering B* **B73** 7-15.

BIOGRAPHICAL SKETCH

Daniel Crunkleton was born on August 9, 1973 to Larry and Dianne Crunkleton in Conway, Arkansas. He grew up in Sand Springs, Oklahoma, where he graduated at the top of his class from Charles Page High School in 1991. He then attended the University of Tulsa on the Marvin J. Bovaird Academic Scholarship, where he received the degree of Bachelor of Science in Chemical Engineering With Honors in 1995. During his undergraduate studies, he held several industrial positions, including as an environmental consultant for Vanguard Environmental Co., Inc., and as a regulatory compliance developer for the Enco Environmental Engineering Co., Inc. He also worked as a full-time research technician for Schlumberger Dowell in their research and development facility. In 1995, he began his graduate studies at the University of Florida, where he received his Ph.D. in May 2002.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy

[illegible]

Timothy J. Anderson, Chairman
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy

[illegible]

Ranganathan Narayanan, Co-Chair
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy

[illegible]

Raj Rajagopalan
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy

[illegible]

Eric D. Wachman
Associate Professor of Materials
Science and Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

May 2002

[illegible]
Pramod P. Khargonekar
Dean, College of Engineering

Winfred M. Phillips
Dean, Graduate School